(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
15 March 2001 (15.03.2001)

PCT

(10) International Publication Number
WO 01/18688 A2

(51) International Patent Classification[7]: G06F 17/30

(21) International Application Number: PCT/US00/11438

(22) International Filing Date: 28 April 2000 (28.04.2000)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:

09/393,390	10 September 1999 (10.09.1999)	US
60/173,807	30 December 1999 (30.12.1999)	US
60/189,969	17 March 2000 (17.03.2000)	US

(71) Applicant: AVANTGO, INC. [US/US]; Suite 300, 1700 South Amphlett Boulevard, San Mateo, CA 94402 (US).

(71) Applicants and
(72) Inventors: KLOBA, David, Douglas [US/US]; 1335 South Montecito, Mountain View, CA 94040 (US). GRAY, Michael, Robert [US/US]; Apartment 3014, 730 Bounty Drive, Foster City, CA 94404 (US). MOORE, David, Marc [US/US]; 23 Baker Street, San Francisco, CA 94117 (US). WHITTAKER, Tom, Edward [US/US]; 2630 Prindle Road, Belmont, CA 94002 (US). WILLIAMS, David, John [AU/US]; 627A Cole Street, San Francisco, CA 94117 (US). WEINSTEIN, Rafael, Zacharia [US/US]; 315 Duncan Street #6, San Francisco, CA 94131 (US). FREEMAN, Josh, Emanuel [US/US]; 4978 Edgar Court #3, San Jose, CA 95118 (US). UPSON, Linus, Michael [US/US]; 1032 8th Avenue, Redwood City, CA 94063 (US).

(74) Agents: LEE, Michael, Q. et al.; Sterne, Kessler, Goldstein & Fox P.L.L.C., Suite 600, 1100 New York Avenue, N.W., Washington, DC 20005-3934 (US).

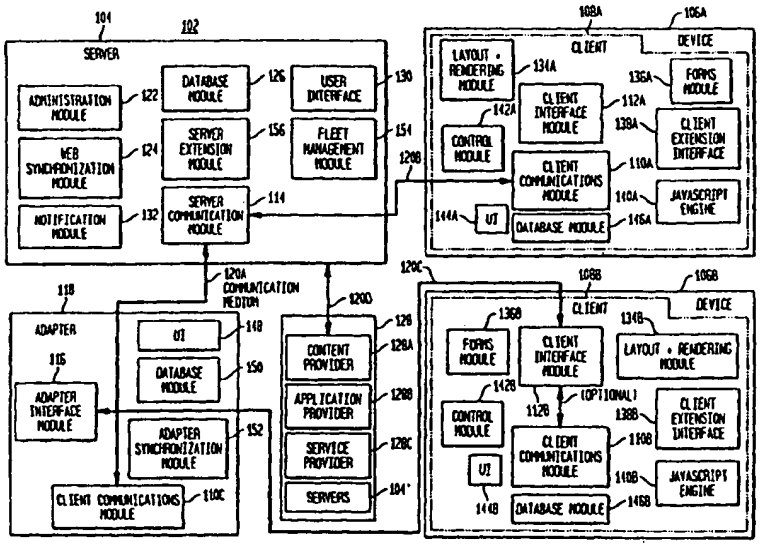
(81) Designated States *(national)*: AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, CA, CH, CN, CR, CU, CZ, DE, DK, DM, DZ, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, TZ, UA, UG, UZ, VN, YU, ZA, ZW.

(84) Designated States *(regional)*: ARIPO patent (GH, GM, KE, LS, MW, SD, SL, SZ, TZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG).

Published:
— *Without international search report and to be republished upon receipt of that report.*

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

(54) Title: SYSTEM, METHOD, AND COMPUTER PROGRAM PRODUCT FOR INTERACTIVE INTERFACING WITH MOBILE DEVICES



(57) Abstract: Described herein are systems, methods, computer program products, and combinations and sub-combinations thereof, for enabling web content (as well as other objects) to be loaded on mobile devices (as well as other types of devices), and for users of mobile devices to operate with such web content on their mobile devices in an interactive manner while in an off-line mode.

WO 01/18688 A2

BEST AVAILABLE COPY

System, Method, and Computer Program Product for Interactive Interfacing with Mobile Devices

Background of the Invention

Field of the Invention

The present invention relates generally to mobile communications, and more particularly relates to technology for using interactive applications while on-line and off-line on mobile devices.

Related Art

A variety of mobile devices (such as personal data assistants, or PDAs) exist. Such mobile devices include ones based on the Palm operating environment and the Windows CE operating environment.

A variety of software applications for those mobile devices also exist.

What does not exist prior to the invention are software applications for enabling web content (as well as other objects) to be loaded on mobile devices, and for users of mobile devices to operate with such web content on their mobile devices in an interactive manner while in an off-line mode.

Summary of the Invention

Briefly stated, the invention includes systems, methods, computer program products, and combinations and sub-combinations thereof for enabling web content (as well as other objects) to be loaded on mobile devices (as well as other types of devices), and for users of mobile devices to operate with such web content on their mobile devices in an interactive manner while in an off-line mode.

These and additional features and advantages of the present invention will become more apparent from the detailed description set forth below when taken in conjunction with the drawings in which like reference characters generally identify corresponding elements throughout.

Brief Description of the Figures

The accompanying drawings, which are incorporated herein and form part of the specification, illustrate embodiments of the present invention and, together with the description, further serve to explain the principles of embodiments of the invention.

FIG. 1A is a block diagram of the invention according to an embodiment of the invention;

FIG. 1B is an alternative block diagram of the invention according to an embodiment of the invention;

FIG. 1B1 is a block diagram of an example data processing unit useful for implementing items from FIGS. 1A and 1B;

FIG. 1C is an example flowchart of a process to interact with objects on a client in an off-line mode according to an embodiment of the invention;

FIG. 1D is an example flowchart of a process to interact with forms on a client according to an embodiment of the invention;

FIG. 1E is an example flowchart of a process to interact with multi-page forms on a client according to an embodiment of the invention;

FIG. 1F1 is an example flowchart of a process for tracking client activity according to an embodiment of the invention;

FIG. 1F2 is an example flowchart of a process for context sensitive processing (such as but not limited to processing relating to advertising) on a client according to an embodiment of the invention;

FIG. 1G is an example flowchart of an initialization process according to an embodiment of the invention;

FIGS. 1H1 and 1H2 collectively illustrate an example flowchart of an "off by N" synchronization process according to an embodiment of the invention;

FIG. 1I1 is an example flowchart of a synchronization process (where the client is connected directly to the server) according to an embodiment of the invention;

FIG. 1I2 is an example flowchart of a synchronization process (where the client is connected to the server via an adapter) according to an embodiment of the invention;

FIG. 1J is an example flowchart relating to server side maintenance of client status information according to an embodiment of the invention;

FIG. 1K is an example flowchart relating to optimizing content for a particular client according to an embodiment of the invention;

FIG. 1L is an example flowchart relating to selectively sending objects to a client depending on whether the client already has the objects according to an embodiment of the invention;

FIG. 1M is an example flowchart relating to syncing channels having collections of objects according to an embodiment of the invention;

FIG. 1N is an example flowchart relating to fleet management according to an embodiment of the invention;

FIG. 1O is an example flowchart relating to automatically adding a channel to the server's collection of channels according to an embodiment of the invention;

FIG. 1P is an example flowchart relating to enabling providers to optimize their objects for use on clients by using predefined meta tags according to an embodiment of the invention;

FIG. 1Q is an example flowchart relating to client side processing of objects based on meta tags contained in the objects according to an embodiment of the invention;

FIG. 1R is an example flowchart relating to server side processing of objects based on meta tags contained in the objects according to an embodiment of the invention;

FIG. 1S is an example flowchart relating to selecting a channel and registering a client, if necessary, according to an embodiment of the invention;

FIG. 1T is an example flowchart relating to processing an anonymous account according to an embodiment of the invention;

FIGS. 1U, 1V, 1W, 1X, 1Y, 1Z, 1AA, and 1AB are used to generally describe embodiments of the invention;

FIG. 2 is an example flowchart of a process to obtain objects from providers according to an embodiment of the invention;

FIG. 3A is an alternative embodiment of a synchronization process;

FIG. 3B is an example block diagram illustrating how XML objects can be served to clients according to an embodiment of the invention;

FIGS. 3C and 4A are views of synchronization processes according to embodiments of the invention;

FIG. 4B is used to indicate how the invention processes hash results according to an embodiment;

FIGS. 5A, 5B, 5C, 5D, 5E, 5F, 5G, 5H, 5I, 5J, 5K, 5L, and 5M relate to user interface functionality according to embodiments of the invention;

FIGS. 6-62 illustrate example screen shots according to embodiments of the invention; and

FIGS. 63A and 63B are event trace diagrams used to describe a synchronization process according to an embodiment of the invention.

It should be understood that these figures depict embodiments of the invention. Variations of these embodiments will be apparent to persons skilled in the relevant art(s) based on the teachings contained herein. For example, the flow charts contained in these figures depict particular operational flows. However, the functions and steps contained in these flow charts can be performed

in other sequences, as will be apparent to persons skilled in the relevant art(s) based on the teachings contained herein.

Detailed Description of the Preferred Embodiments

1. Overview of Embodiments of the Present Invention

Embodiments of the present invention are briefly described in this section.

Briefly stated, the invention is directed to placing objects such as, but not limited to, Internet or Web content on data processing devices, such as but not limited to mobile devices. Table 1 lists examples of such Internet content, although the invention is not limited to these examples.

TABLE 1. Internet Content

Internet content includes but is not limited to:
HTML
JavaScript™
Channels
Java™
ActiveX
Multimedia:
Images (e.g., JPEG, GIF, PNG, vector graphics, etc.)
Audio Files (e.g. MP3)
Video (e.g. AVI)
Streaming Content: Voice/Data/Video
Binary files
XML
Applications
Data Objects
Documents

Anything that can be delivered via a "browser"

Table 2 lists examples of mobile devices, although the invention is not limited to these examples.

TABLE 2. Mobile Devices

Mobile devices include but are not limited to:
Handheld Computers
Cellular Phones
Internet-enabled Phones
Pagers
Radios
TVs
Audio Devices
Car Audio Systems
Recorders
Text-to-Speech Devices
Bar-code Scanners
Net Appliances
Mini-browsers
Personal Data Assistants (PDAs)

FIG. 1U illustrates the concept of the invention of placing objects on data processing devices, such as mobile devices.

1.1. Enabling Mobile Devices to Interact With Networked Applications

The invention includes technology for using applications on mobile devices that interact with the Internet or with intranets. The invention enables

applications available via a network or via an Internet/intranet to download and to run on mobile devices. Consequently, the invention includes software and methods for administering a server that manages the variables relevant to a mobile device/server environment.

The invention enables:

Mobile devices to operate in conjunction with a Web server, even when the mobile devices are not coupled directly to the PC using portable on-device servers: Web pages are loaded, viewed, cached, and deleted even when the device is not coupled to any network.

Mobile devices to operate in conjunction with the Web, Internet, or intranet via a connection mechanism and then in disconnected mode or with the Web, Internet, or intranet in wireless mode with a continuous or a discontinuous connection mechanism.

A technique for interactive connectivity between handheld computers and computer networks.

Fleet management for centrally administering information in a handheld network environment that includes, but is not limited to, user data, user groups, group channels, channel data, personal channels, commercial channels, user accounts, corporate account, software groupings, personal information management, form delivery, form management, device configuration, device databases, device contents, and devices parameters.

Obtaining updated Web pages and other network objects, for use when the mobile device is not communicating with the PC.

An example mobile device/server environment is shown in FIG. 1V.

1.2. Rapid Transfer of Web Pages to Mobile Devices

To improve efficiency of data exchange between mobile devices and networked content, the invention includes an improved communication protocol that collects requests and responses for network objects into a smaller number of protocol (such as HTTP) requests and responses. The server also determines the nature and the resources of the mobile device. This protocol is represented, for example, in FIG. 1W.

Downstream, the data is encoded in a data format called ABC (tokenized version of the data) and sent to the device. Already Been Chewed (ABC) format creates a tokenized codification of HTML pages that is sent to the device. (The device receives the ABC and presents the material on the device.)

The HTML page is encoded into ABC and sent to the device. The encoding is a mapping of parent and child HTML elements and/or resources to alphanumeric values.

The sync operation of the invention includes various synchronization processes that can collect information from the Internet to a server, and to the client. The usage of the term "sync," as described herein, refers to the overall operation of connecting a client to a server for the exchange, interaction, creation, and removal of data.

In one embodiment, syncing can be defined as mirroring data on a client and a server, such that the data is the same on client and server. In other embodiments, syncing can be defined as overwriting data on a client or on a server, such that the data on either a client replaces the data on a server, and vice versa.

In one embodiment, a sync operation involves a user placing a mobile device into an adapter that includes a sync button. The adapter is connected to a server. Upon pressing the sync button, the user initiates the sync operations of the present invention, which include various synchronization processes (specific delivery modes). Thus, the term sync is meant to refer to the overall operation

of linking a client to a server. Synchronization is meant to refer to the specific process of copying, adding, filtering, removing, updating and merging the information between a client and a server. Any number of synchronization processes can be executed during a sync.

Before being sent downstream the data is compared to the data that is known to be on the client and then the client is updated all at once in a one-up/one-down synchronization method, which is represented in FIG. 1X. The server sets the client to preemptively prepare all device information necessary during the sync. Then the server receives the set of information in a one-up fashion. The server collates the information and sends the information in a one-down fashion. This optimizes the sync's efficiency and speed. The sync process is represented in FIGS. 1Y and 1Z.

1.3. Optimizing Content of Web Pages for Mobile Devices

When Web content and other network objects pass through the server they are processed to minimize their size and to optimize their delivery to mobile devices: for presentation, for ease of use, for efficiency, for size, etc.

The invention uses server logic to optimize content. The server assesses the mobile device to optimize web content for the device. Factors that the server logic considers when performing this optimization include, but are not limited to:

Dynamic memory specifications

High memory specifications

Protected Memory

Storage Memory

Database Memory

Available storage space

Screen size

User profile(s)

Color depth

Applications on device

Buttons on-device

Data markers (e.g., cookies, tokens)

Preferences

Fonts

Font specifications

Sync type

Synchronization types

Supported data types

Supported mime types

Connection/Network profile

An example optimization process is shown in FIG. 1AA.

On the server, the graphic is optimized per the state information of the device. If the device sends down the need for the graphic on a page for a device with a display that is 27 cm wide and in grayscale, the server sends its best version of a graphic optimized for that environment.

The technology of the invention is extended by tags on HTML pages that identify content that is designed for additional modifications. Any and all bytes processed by the server are potentially examined for compression/optimization. The server detects the tag and executes the necessary logic.

Table 3 illustrates example tags (the invention is not limited to the tags shown in Table 3).

TABLE 3. Sample Markup Language

Tag	Effect
<META NAME="Handheld-Friendly" content="True">	This tag enables several HTML features that are normally turned off. Most notably, The invention does not try to display TABLE tags or the HSPACE and VSPACE attributes of IMG tags unless the page is marked as "HandheldFriendly". Most TABLEs or H/VSPACEs are designed for much larger screens.
<AGIGNORE></AGIGNORE>	Used in a wireless channel. Use the AGIGNORE tag to surround content within an HTML page that may be inappropriate or unattractive on Internet-enabled phones.
<AGPAGEBREAK TITLE="your title">	Used in a wireless channel. Breaks up pages on request. When processing pages for devices other than WAP phones, the server ignores the AGPAGEBREAK tag.

Web Content Aggregation, Web Channel Development, and Web Content Delivery for Users of the Internet and of Mobile Devices

The invention is extended by the coupling of devices to the content available at the server web site (see the example shown in FIG. 1AB).

These and other embodiments of the present invention are described in greater detail below.

Structural Embodiments of the Present Invention

FIG. 1A is a block diagram of a data processing environment 102 according to an embodiment of the invention. The data processing environment 102 includes a server 104 (although only one server 104 is shown, in practice the data processing environment 102 may include a plurality of servers), one or more devices 106, one or more adapters 118, and one or more providers 128.

Generally, the server 104 maintains a collection of channels. In an embodiment, a channel comprises a collection of objects. An object is any entity that can be transferred to a client 108, such as but not limited to content, applications, services, images, movies, music, links, etc.

A channel includes a number of properties. At least some of these properties define the objects that the channel includes. Such properties include, but are not limited to, the following:

A name of the channel.

A location of a root object (such as but not limited to a URL). In an embodiment, this root object is included in the channel.An indication of the number of levels below the root object, for which to include objects in the channel. For example, in an embodiment, if this property is equal to "1 level," then all objects that are 1 level down from the root object (reached by traversing links in the root object), are included in the channel. If this property is equal to "2 levels," then all objects that are 1 level down from the root object (reached by traversing links in the root object), and all objects that are 1 level down from those objects (reached by traversing links in those objects), are included in the channel. Embodiments of the invention allow "uneven" trees, where some branches of the tree extent to a greater number of levels than other branches of the tree. In other embodiments, the trees are even or balanced.

A maximum size of the channel. For example, if this is set to 500 Kbytes, then the aggregate size of the objects in the channel cannot be greater than 500 Kbytes. If the aggregate size of the objects in the channel is greater than this value, then embodiments of the invention may delete objects from the channel and/or delete portions of objects in the channel.

An indication of which resource objects are enabled for the channel:

An indication of whether or not images are to be included in or excluded from objects in the channel; and

An indication of whether or not scripts are enabled in objects in the channel.

A refresh methodology.

It is noted that the properties associated with channels may vary from implementation to implementation. Also, implementations may employ combinations of the above properties, and/or properties in addition to the following, as will be appreciated by persons skilled in the relevant art(s).

The invention includes processes for managing channels, including but not limited to adding channels to the collection of channels maintained by the server 104.

The server 104 offers channels to clients 108. A client 108 may access the server 104 and view the collection of channels. The client 108 may then select any combination of the channels in the collection. The server 104 maintains a list of the channels associated with each of the clients 108.

During a synchronization process, the server 104 loads a device 108 with the channels associated with the client 108. Generally, the server 104 does this by obtaining from providers 128 the objects defined by the channels, and causing those objects to be stored on the client 108. Thus, during the synchronization process, the server 104 will load the client 108 with the selected channels. More particularly, the server 104 will load the client 108 with the objects associated with the channels.

The client 108 may process and use those objects when not connected to the server 104. The invention enables the client 108 to actively interact with the objects and channels.

In one embodiment, the client 108A directly interacts with the server 104 via some transmission medium 120B, which may be any wired or wireless medium using any communication protocol.

In another embodiment, the client 108B indirectly interacts with the server 104 via an adapter 118. For example, the client 108B may be a mobile device (such as a Palm device) and the adapter 118 may be a cradle and a computer coupled to the cradle (the mobile device is inserted into the cradle). In this instance, the adapter 118 presents itself to the server 104 as a client 108B (via client communications module 110C). When the server 104 sends objects to the adapter 118, the adapter interface module 116 writes those objects to client 108B. In embodiments, adapter interface module 116 can be a Hot Sync™ Manager, an Active Sync™, etc. It is noted that the invention is not limited to any of the implementation examples discussed herein.

The components shown in FIG. 1A shall now be described in greater detail.

The server 104 includes an administration module 122, a database module 126, a user interface 130, a web synchronization module 124, a server extension module 156, a fleet management module 154, a notification module 132, and a server communication module 114. Other embodiments of server 104 may include a subset of these modules, and/or may include additional modules.

The administration module 122 controls and manages the states of the server 104 and the clients 108. For example, the administration module 122 manages and controls groups of clients 108, permissions assigned to clients 108, groups, and channels. For example, the administration module 122 administers the users/clients 108 assigned to groups, and the channels associated with users. These and additional functions performed by the administration module 122 are described herein.

The database module 126 controls access to databases associated with the server 104. The database module 126 maintains information relevant to the clients 108, as well as information relevant to the modules contained in the server

104. The database module 126 manages information on the collection of channels maintained by server 104. These and additional functions performed by the database module 126 are described herein.

The user interface 130 is, in an embodiment, a graphical user interface (GUI) that enables users and clients 108 to access functions and modules offered by the server 104. More generally, the user interface 130 within server 104 provides access to server 104 and the modules and resources contained therein.

The invention supports various server web sites that are available through any communication medium, such as but not limited to the Internet, intranets, direct dial up links, etc. The UI 130 enables such web sites.

These and additional functions performed by the user interface 130 are described herein.

The web synchronization module 124 is an application/instance of server extension module 156, and controls synchronization of web content to client 108. The invention may include other synchronization modules (which are application/ instances of server extension module 156) that control synchronization of other types of objects to clients 108. For example, the server 104 may administer a calendar that may be installed on clients 108. The synchronization of appointments, events and/or dates on this calendar between clients 108 and the server 104 may be performed by a calendar synchronization module. These and additional functions performed by the server extension module 156 are described herein.

The fleet management module 154 performs functions associated with fleets of clients 108, which are groups of clients 108. For example, fleet management module 154 may perform global or mass operations on groups (fleets) of clients 108, such as loading or updating an application on groups (fleets) of clients 108. Another example of a mass operation is retrieval of information on clients 108 in a fleet, such as the free memory in clients 108 in a fleet (this would help an organization determine if its clients 108 need a memory

upgrade). These and additional functions performed by the fleet management module 154 are described herein.

The server extension interface/module 156 enables modules, such as third party modules, to operate in or work with the server 104 (and modules contained in the server 104). The server extension module 156 presents an API (application programming interface). Modules in the server 104 may operate with other devices in the server 104 by conforming to the server API.

For example, the web synchronization module 124 and the fleet management module 154 (as well as other types of synchronization modules, not shown in FIG. 1A) may interact with databases on the server 104 via the database module 126 by going through the server extension module 156. The web synchronization module 124 and the fleet management module 154 may not be able to interact directly with the database module 126 for a number of reasons. For example, they may support different data formats, or simply "speak different languages." However, they can interact via the server extension module 156 as well as other server modules as long as they conform to the API of the server extension module 156. This is true of any modules in the server 104, or that interact with the server 104.

Server communication module 114 enables communication between the server 104 and entities external to the server 104, such as clients 108, adapters 118, providers 128, work stations, etc. The server 104 communicates with these entities via communication mediums 120, which may be any type of wireless or wired communication using any protocol. It is noted that multiple server communication modules 114 may execute in a single server 104. For example, in one embodiment, server communication module 114 is a TCP/IP stack. In another embodiment, server communication module 114 is a secure socket layer stack or a compression stack. The invention is not limited to any implementation examples discussed herein. These and additional functions performed by the server communication module 114 are described herein.

The notification module 132 sends objects to clients 108 beyond objects related to channels associated with clients 108. Such objects could be requested by client 108 in advance. For example, a client 108 could ask for a notification when an event happens, such as when a stock reaches a target price. When the event occurs, the notification module 132 would cause an appropriate notification(s)/object(s) to be sent to the client 108. Alternatively, the notification module 132 may send objects to clients 108 without any prior explicit request from the client 108. For example, the notification module 132 might send channels to clients 108 when such channels are identified to be similar to those already selected by the clients 108. Also, the notification module 132 might send appropriate notifications/objects to the clients 108 when such clients 108 receive email or faxes at the server 104. In embodiments, the notification module 132 transmits such objects to the client 108 immediately when the event occurs, during the next synchronization with the client 108, or at some other future synchronization.

An alternative representation of server 104 is shown in FIG. 1B. FIG. 1B illustrates, for example, that messages from entities outside of server 104 are received by server extension interface/module 156 via server communications modules 114. Generally, such messages represent requests for the server 104 to perform various functions. The server extension module 156 conceptually operates as a dispatcher who routes such messages to other modules contained in the server 104, such as web synchronization module 124 (who handles requests to synchronize with web content), notification module 132, fleet management module 154 (who handles fleet related requests), and/or third party modules 155 (such as other synchronization modules). Thus, the invention supports modules 155 generated by third parties to perform various functions. Such modules 155 "plug-in" to the server 104 via the server extension module 156.

Referring again to FIG. 1, the devices 106 may be any type of data processing device. In embodiments of the invention, the devices 106 are mobile computing devices, although the invention is not limited to these embodiments.

In such example embodiments, the devices 106 may include, but are not limited to, handheld computers, cellular phones, internet-enabled phones, pagers, radios, tvs, audio devices, car audio systems, recorders, text-to-speech devices, bar-code scanners, net appliances, mini-browsers, personal data assistants (PDAs), etc.

In embodiments of the invention, the devices 106 include software, hardware, and/or combinations thereof related to client functionality (such client functionality is described herein). When a device 106 includes such software, hardware, and/or combinations thereof, the device 106 is referred to herein as a client 108. Accordingly, it can be said that the data processing environment 102 includes one or more clients 108.

Clients 108 each may include a layout and rendering module 134, a forms module 136, a control module 142, a user interface 144, a client extension interface 138, a client interface module 112, a client communications module 110, a JavaScript™ engine 140, and a database module 146. Other embodiments of clients 108 may include a subset of these modules, and/or may include additional modules.

Layout and rendering module 134 controls the processing of data objects on client 108, such as the layout and rendering of data objects on client 108. For example, the layout portion of module 134 obtains information from databases of the client 108 (via the database manager 146) and determines where such information should be rendered on the display of the client 108. Such information may include anything that can be rendered, such as but not limited to images, text, links, etc. The rendering portion of module 134 is responsible for drawing items on the display (drawing bits to the screen). These and additional functions performed by the layout and rendering module 134 are described herein.

The forms module 136 controls and manages forms. For example, in embodiments the forms module 136 manages aspects of off-line forms, such as HTML forms and/or multi-page forms. The forms module 136 enables access to and user interaction with forms (in some embodiments, the forms module 136 via UI 144 enables users of client 108 to directly access forms). The forms module

136 maintains the status of forms. Forms module 136 can also include a forms manager (not shown) to provide added functionality. These and additional functions performed by the forms module 136 are described herein.

The user interface 144 is preferably a graphical user interface that enables users to interact with client 108 and functions and modules provided by the client 108. More generally, UI 144 controls how functions presented by modules of the client 108 are presented to users. The UI 144 controls how users interact with such functions and modules. It is noted that the functionality of the UI 144 may be distributed. For example, portions of the UI 144 may reside in the forms module 136, as well as other modules of client 108. These and additional functions performed by the user interface 144 are described herein.

The client extension interface 138 enables modules, such as third party modules, to operate in or work with the client 108 (and modules contained in the client 108). The client extension interface 138, also known as an on-device server, presents an API (application programming interface) that is, in embodiments, common to clients 108 on many architectures.

Modules in the client 108 can work together via the client extension interface 138. For example, the JavaScript™ engine 140 may decide that it wishes to display a message to the user. To do this, the JavaScript™ engine 140 would work through the client extension interface 138 to cause the UI 144 to display the message to the user. The JavaScript™ engine 140 may not know how to directly interact with the UI 144. However, as long as both the JavaScript™ engine 140 and the UI 144 conform to the API of the client extension interface 138, then they can operate together.

Similarly, the control module 142 may decide that it needs to store some data in a database. The control module 142 would do this by working with the client extension interface 138 to access the database module 146 to effect such a modification to the databases in the client 108. These and additional functions performed by the client extension interface 138 are described herein.

The JavaScript™ engine 140 executes objects written in the JavaScript™ language that operate on client 108. As noted, the JavaScript™ engine 140 conforms to the API of the client extension interface 138, and works with the client extension interface 138 to work with other modules in client 108. These and additional functions performed by the JavaScript™ engine 140 are described herein.

Although not shown in FIG. 1A, embodiments of the invention include other engines for executing other types of scripts on client 108. These other engines can interact with other modules on client 108 as long as the engines conform to the API of the client extension interface 138.

The database module 146 controls access to databases associated with client 108. More generally, the database manager 146 controls access to resources on the client 108. For example, the control module 142 may interact with the database manager 146 to open an address book in the databases, and to write a record to the address book. Alternatively, the forms module 136 can interact with the database module 146 to access forms that are stored in the databases. These and additional functions performed by the database module 146 are described herein.

Client communications module 110 enables the client 108 to interact with external entities, such as server 104. In embodiments, the client communications module 110 enables TCP/IP traffic, although the invention is not limited to this example. More generally, the client communications module 110 enables communication over any type of communication medium 120, such as wireless, wired, etc., using any communication protocol, such as a pager protocol. These and additional functions performed by the client communications module 110 are described herein. The client interface module 112 enables the client 108 to communicate with adapters 118. Client interface module 112 optionally links to client communications module 110 in some embodiments to provide functionality (for example, when the client communications module 110 uses a wireless modem's drivers, which are accessed via client interface module 112). In

embodiments, the client interface module 112 may be Hot Sync™ Manager in the Palm operating environment, or Active Sync™ in the Windows CE™ operating environment, or Pilot Link™ in the Unix operating environment. It is noted that these implementation examples are provided for illustrative purposes only. The invention is not limited to these examples. These and additional functions performed by the client interface module 112 are described herein.

The control module 142 coordinates the activities of the other modules in client 108 so that all the modules share resources properly. For instance, control module 142 can determine priorities for shared resources such as processing time, accessing memory, etc.

Providers 128 are sources of various types of objects, such as but not limited to content (content providers 128A), applications (application providers 128B), services (service providers 128C), etc. Providers 128 may also include servers 104' (similar to server 104), which may provide objects such as but not limited to content, applications, services, etc. For example, and without limitation, the application providers 128B may provide objects relating to (without limitation) operating system updates/changes, system upgrades, application updates/changes, etc.

Adapters 118 include an adapter interface module 116, a user interface 148, a database module 150, an adapter synchronization module 152, and a client communications module 110. Other embodiments of adapters 118 may include a subset of these modules, and/or may include additional modules.

Client communications module 110 is the same as similarly named modules in clients 108.

The adapter interface module 116 enables the adapter 118 to communicate with clients 108.

The adapter synchronization module 152 is involved with synchronization operations between server 104 and clients 108.

The UI 148 enables users to interact with modules and functions of adapter 118.

The database module 150 controls access to databases associated with adapter 118. The database module 150 manages information needed for clients 108 to remain in sync with server 104. In some embodiments, the adapter 118 does not include the database module 150 or the UI 148 (i.e., in embodiments where the adapter 118 operates essentially as a pipe, as in some embodiments on Unix).

These and additional functions performed by modules of the adapter 118 are described herein.

2.1. Example Implementation Embodiments

FIG. 1B1 illustrates a block diagram of a data processing unit 103A that can be used to implement the entities shown in FIGS. 1A and 1B. It is noted that the entities shown in FIGS. 1A and 1B may be implemented using any number of data processing units 103A, and the configuration actually used is implementation specific.

Data processing unit 103A may represent laptop computers, hand held computers, lap top computers, and/or any other type of data processing devices. Which type of data processing device used to implement entities shown in FIGS. 1A and 1B is implementation specific.

Data processing unit 103A includes a communication medium 103B (such as a bus, for example) to which other modules are attached.

Data processing unit 103A includes one or more processor(s) 103C, and a main memory 103D. Main memory 103D may be RAM, ROM, or any other memory type, or combinations thereof.

Data processing unit 103A may include secondary storage devices 103E, such as but not limited to hard drives 103F or computer program product interfaces 103G. Computer program product interfaces 103G are devices that access objects (such as information and/or software) stored in computer program products 103. Examples of computer program product interfaces 103G include,

but are not limited to, floppy drives, ZIP™ drives, JAZ™ drives, optical storage devices, etc. Examples of computer program products 103H include, but are not limited to, floppy disks, ZIP™ and JAZ™ disks, memory sticks, memory cards, or any other medium on which objects may be stored.

The computer program products 103H include computer useable mediums in which objects may be stored, such as but not limited to optical mediums, magnetic mediums, etc.

Control logic or software may be stored in main memory 103D, secondary storage device(s) 103E, and/or computer program products 103H.

More generally, the term "computer program product" refers to any device in which control logic (software) is stored, so in this context a computer program product could be any memory device having control logic stored therein. The invention is directed to computer program products having stored therein software that enables a computer/processor to perform functions of the invention as described herein.

The data processing unit 103A may also include an interface 103J which may receive objects (such as data, applications, software, images, etc.) from external entities 103N via any communication mediums including wired and wireless communication mediums. In such cases, the objects 103L are transported between external entities 103N and interface 103J via signals 103K, 103M. In other words, such signals 103K, 103M include or represent control logic for enabling a processor or computer to perform functions of the invention. According to embodiments of the invention, such signals 103K, 103M are also considered to be computer program products, and the invention is directed to such computer program products.

3. Operational Embodiments of the Present Invention

3.1. Enabling On-Device Servers, Off-Line Forms, and Dynamic Ad Tracking On Mobile Devices

3.1.1. Caching Objects on Clients for Off-Line Browsing

Referring to FIG. 1C, a flowchart 160 is shown that illustrates a process by which content is captured and stored on client 108 to thereby allow the user to view the content on device 106 offline, according to an embodiment of the invention. Flowchart 160 begins with a user expressing the desire to see content on device 106 (step 160A). Device 106 may be a handheld unit of the type as described herein.

It is noted that when client 108 is resident on device 106, the terms client and device are used interchangeably herein (unless noted otherwise either explicitly or implicitly by context).

For convenience, functions are described herein as being performed by certain module(s). The invention is not limited to these descriptions. In embodiments, such functions are performed by other module(s). This is true throughout the discussion herein.

While device 106 is described in terms of the above-mentioned units, this is for convenience only and is not intended to limit its application. In fact, after reading the following description, it will be apparent to one skilled in the relevant art(s) how to implement the following invention in alternative embodiments (e.g., by providing the functionality of device 106 in emulation on a desktop PC or workstation).

In manipulating device 106, the user interacts with server 104 via user interface 130 to identify channels (step 160B). In one embodiment of the invention, channels contain content. As previously mentioned, content can be information. Additionally, content may be organized topically into areas of interest to a user. Generally, the channels can include any objects.

In another embodiment of the invention, the content in channels may be altered over time. For example, channels may be updated periodically in a predetermined fashion. In another example, channels are updated conditionally

upon the occurrence of an event. In order to obtain the altered content, the user synchronizes device 106 via server 104 (step 160C). The synchronization process is discussed in detail in later sections of this application. During synchronization, server 104 gathers channel content and sends it to device 106 (step 160D).

3.1.2. Channel Aggregation and Selection By Clients

As discussed herein, the server 104 collects or aggregates channels for selection by clients 108.

FIG. 2 is a flow diagram describing in further detail the process 160D for gathering channels and sending the channels to device 106 according to an embodiment of the invention. The process begins with step 202.

In step 202, top level resources that server 104 needs to fulfill client 108's request are identified by server 104. For example, if client 108 is requesting a full synchronization, server 104 will identify any changed objects from providers 128 and send them to client 108. Client 108 can also request that a subset of providers 128 be updated Server 104 will identify any changed objects within the subset of providers 128 and send them to client 108.

In step 204, Web synchronization module 124 communicates with providers 128 to obtain the top-level resources. Other resources, such as images, links, JavaScript™, etc., needed to maintain the integrity of the information provided for each object are then determined in step 206. In step 207, objects are transformed so that they fit within the parameters of device 106. Such parameters may include, but are not limited to, memory size, the size of device 106, capabilities of device 106, etc. When all resources have been amassed to fulfill client 108's request, the process proceeds to step 208.

In step 208, the objects retrieved in the preceding steps are compared with the objects already cached on device 106. Server 104 determines the set of changes that have occurred between the retrieved objects and the objects already cached on device 106 in step 210. Only the set of changes determined in step 210

are transmitted to device 106.To improve the efficiency of the synchronization session between client 108 and server 104, as well as optimize the content displayed by client 108, a communication protocol collects requests and responses for network objects into a smaller number of protocol (such as HTTP) requests and responses. In an embodiment of the present invention, specific values are transformed in a conversion process to a tokenized encoding that is optimized for the device, client, and/or application. In one example, the encoding can be a mapping of parent and child HTML elements and/or resources to alphanumeric values designed to present content on the client's display.

FIG. 1W illustrates a block diagram of one embodiment of the optimized downstream protocol. FIG. 1W illustrates raw objects from provider 128, server 104, and device 106. Server 104 transforms the raw objects into an efficient representation for displaying the objects on device 106. For example, HTML objects are transformed into a tokenized compressed version of HTML. In another example, resources such as images, JavaScript™, etc. are transformed into tokenized compressed versions of resources. Generally, "human friendly" HTML is transformed into "machine friendly" format that is compact and regular (thereby reducing the requirements on the client 108 to process the objects). During the synchronization session, server 104 also determines the nature and the resources of device 106. Thus, server 104 can determine the amount of content to download to device 106 as well as the features of device 106. For example, device 106 may or may not be able to display color graphics and text. Therefore, a gif image would be scaled to fit the screen size of client 108 as well as reducing the color to a black and white image.

Returning again to FIG. 1C, the revised channels are cached on device 106 so that the content can be later accessed (step 160E) by the user in an off-line manner.

In order to access the cached content, the user launches client 108 on device 106 (step 160F). The user selects channels via user interface 144 (step 160G). User interface 144 provides logic for displaying the means to access the

resources of client 108. In one embodiment, user interface 144 displays a list of channels from which a user may select specific channels.

Once a channel is selected, layout and rendering module 134 displays the selected channel (step 160H). In one embodiment, the content of the selected channel is presented. In another embodiment, a form is presented for a user to enter a query. Thus, the invention allows the user to interact with the channels (step 160I) even when not connected to server 104 or provider(s) 128. In one embodiment, a user is essentially viewing Internet content off-line via cached Web pages.

3.1.3. Forms to Enable Off-Line Interactive Processing By Clients

As described herein, in one embodiment channel content may contain a form or forms. For a single form, the form may be a multiple submit form or a single submit form. A multiple submit form contains multiple submissions for a single page. A single submit form contains one submission for a single page. Multiple submit forms will allow a user to submit the form multiple times prior to synchronization. Alternatively, the single submit form can only be submitted once per synchronization. Referring to FIG. 1D, flowchart 160I illustrates a user's interaction with a channel having a single form (FIG. 1D is an example embodiment of step 160I in FIG. 1C). Starting with step 162A, a page is displayed by user interface 144 that contains form elements. Form elements may contain fields for the entry of data/commands such as query criteria. For example, query criteria may include identification information, location information, etc. Additionally, form elements may present a user with a list of choices and means by which a choice can be selected, such as radio boxes, check boxes, popup menus, etc. A user enters data/commands into the form elements via user interface 144 (step 162B).

Forms module 136 caches the data/commands for later synchronization (step 162C). During the synchronization process, which is discussed below in detail, control module 142 provides an appropriate notification (step 162D). In one embodiment, the appropriate notification is a message displayed by control module 142 that the response to the form will be obtained during the next synchronization.

Forms module 136 maintains the status of the forms cached and manages the completion of the forms (step 162E). In one embodiment, a user can access forms module 136 directly and review the cached forms before and after synchronization.

Multiple page forms may also be implemented. Multiple page forms may result from a single form that is too large to display on client 108. In this instance, server 104 transforms the single page form into multiple page forms for display on client 108. Referring to FIG. 1E, flowchart 160I' illustrates user interaction with a multiple page form (FIG. 1E is an embodiment of step 160I in FIG. 1C). Starting with step 164A, the user accesses a channel containing a multiple page form. User interface 144 displays the first page of the form (step 164B). In much the same way as in step 162B of FIG. 1D, the user enters data/commands into the form elements on the page of the displayed form (step 164C).

Client extension interface 138 stores the data/commands from the displayed form page (step 164D). User interface 144 displays the next page of the form (step 164E). Steps 164C, 164D, and 164E are repeated until all the pages of the form are completed. In one embodiment, client extension interface 138 delivers the completed multiple page form as a single form to forms module 136 (step 164F). In another embodiment, client extension interface 138 delivers each completed page of the form to the forms module 136 (not shown). During the synchronization process, which is discussed below in detail, control module 142 provides an appropriate notification (step 164G). In one embodiment, the appropriate notification is a message displayed by control module 142 that the

response to the form will be obtained during the next synchronization. Similarly to step 162E of FIG. 1D, forms module 136 maintains and manages the forms (step 164H).

3.1.4. Tracking Client Behavior

As described herein, the invention enables client 108 to record user/client behavior. Examples may include, but are not limited to, tracking page impressions, such as tracking the number of times that a particular user has viewed a particular page or listened to a particular song, the amount of time a user spends viewing a page, or any other client activity. Other information that can be tracked includes, but is not limited to, user name, current time of request, page that is being viewed, the referred page, etc.

FIG. 1F1 is a flow diagram describing a method for tracking page impressions offline, and for tracking other client 108 activity. In step 166A, the user selects a page to view. It is then determined whether provider 128 of the page/object has requested that client activity be tracked and recorded (step 166B). If provider 128 has not requested that client activity be recorded, the page is displayed and the client is not tracked (step 166C). If provider 128 has requested that client activity be tracked and recorded, the process proceeds to step 166D.

In step 166D, client extension interface 138 tracks client activity (as defined by the provider 128). The process proceeds to step 166E.

In step 166E, the tracked information is transmitted to server 104 upon synchronization. In step 166F, server 104 then sends the information to the appropriate provider 128. The provider 128 may pay some compensation for this service.

3.1.5. Display of Context Sensitive Objects

The invention enables the processing of context sensitive objects based on context sensitive triggers while the client 108 is browsing pages/objects in either an off-line mode (i.e., when not connected to server 104) or an on-line mode (i.e., when connected to server 104). Table 4 displays a listing of exemplary context sensitive objects. Table 5 displays a listing of exemplary context sensitive triggers. One skilled in the relevant art(s) would realize that other context sensitive objects and context sensitive triggers may be used without departing from the scope of the present invention. This process is shown in FIG. 1F2.

TABLE 4

Context Sensitive Objects
Business card
Advertisement
e-mail
to do list
calendar event
ticket notification
channels

TABLE 5

Context Sensitive Triggers
Global positioning satellite locator
Zip code
Time of day
User preferences
Last sync location
In range of a transmitter (e.g., bluetooth)

Credit card

Temperature

Altitude

Agent, arbiter, avatar

In step 167A, the user selects a page to view on the client 108. The page is obtained from the cache of the client 108, or if not in the cache then from the server 104 (in on-line mode, or via the sync process when not on-line).

In step 167B, the client 108 determines if there are any context sensitive objects. Such objects may be related to the page of step 167A, or status information of client 108, or a combination thereof (or sensitive to other factors, as will be appreciated to persons skilled in the relevant art(s)).

If there are not context sensitive objects, then in step 167D the page is displayed on client 108.

If there are context sensitive objects, then in step 167C the objects are processed and the page is displayed on client 108. Processing of the objects depends on the nature of the objects. For example, if the object is an image, then the image is displayed. If the object is a script, then the script is processed.

In an embodiment, the objects may be advertisements, although the invention is not limited to this example. The sources of objects may pay the server 104 (or a party associated with server 104) for the ability to have such objects loaded and processed on clients 108.

3.2. Syncing to Mobile Devices

Referring to FIG. 1G, flowchart 168 illustrates a synchronization initialization process according to an embodiment of the present invention. This process is also explained by a corresponding example event trace diagram in FIGS. 63A and 63B.

Client 108 initializes a synchronization session and sends a null data marker [Cn] to server 104 (step 168A). See 6304 in FIG. 63A. In one embodiment, a data marker is a synchronization token associated with the state of data on the client 108. More specifically, in one example, a synchronization token is a number that is sequentially increased by server 104 with each synchronization. Server 104 tells client 108 which client databases it wishes to track and sends data marker [C1] to client 108 (step 168B). See 6306 in FIG. 63A. At this point, the synchronization data marker for the client 108 is equal to C1 at both the client 108 and the server 104, as indicated by 6308 and 6310, respectively.

FIGS. 1H1 and 1H2 collectively illustrate a synchronization process that occurs subsequent to the initialization process of FIG. 1G.

As shown in flowcharts 170 in FIG. 1H1, and 170' in FIG. 1H2, the synchronization process checks to see if it can proceed from an earlier known state of information on the client. In one embodiment, client communication module 110 of client 108 initializes a synchronization session (step 170A). Client control module 110 of client 108 sends a current data marker C1 to web synchronization module 124 on server 104 (step 170B). This is indicated by 6320 in FIG. 63B.

Server 104 uses the data marker C1 received from client 108 (6320 in FIG. 63B) to determine whether the last synchronization with client 108 was successful (step 170C). In an embodiment, a successful synchronization is indicated if the value of the synchronization data marker that is maintained by the server 104 for the client 108 is equal to the data marker sent by the client 108 to the server 104 in the sync request. In the example of FIG. 63A, the data marker sent by the client 108 to the server 104 in the sync request is C1 (6320 in FIG. 63B), which matches the data marker maintained in the server 104 for the client 108 (6310 in FIG. 63B). Accordingly, in the example of FIG. 63B, the server 104 in step 170D determines that the last sync with the client 108 was successful.

Accordingly, in step 170E, a normal sync process is performed, which is described below.

If the last sync was not successful as determined in step 170D, then control flows to step 170F (described below). FIG. 63B shows an example where the sync with client 108 is not successful. At 6324, the client 108 sends a sync request with data marker C2. At this point in time, the data marker maintained by the server 104 for the client 108 is equal to C2 (6312 in FIG. 63B). Accordingly, a match exists, and in 6326 the server 104 performs a normal sync and transmits new data marker C3 to client 108. However, due to some event 6328, this transmission is not received by client 108. Thus, client 108 never receives the new data marker C3. When the client 108 sends the next sync request, it transmits data marker C2 (6330 in FIG. 63B). At this point in time, the data marker maintained by the server 104 for the client 108 is equal to C3 (6316 in FIG. 63B), which does not match data marker C2 received from the client 108 in the sync request (6330 in FIG. 63B). Thus, the server 104 in step 170D determines that the last sync with client 108 was not successful. Accordingly, step 170F is performed.

In step 170F, the server 104 compares the latest data marker received from the client 108 (C2 in the example of 63B) with ones stored in the server 104 for the client 108. Essentially, the server 104 attempts in step 170F to "roll back" to a previous known state of client 108. In the example of FIG. 63B, the server 104 in step 170F determines that it can roll back to a known state of the client 108 corresponding to data marker C2 (6312 in FIG. 63B).

In steps 170G, 170H, and 170I, the server 104 determines what instructions are needed to cause the client 108 to roll back to the known state associated with data marker C2 identified in step 170F, and what instructions are needed to cause the client 108 to move forward from the previous state associated with data marker C2 to the current state associated with data marker C3.

In steps 170J, the instructions determined from steps 170G, 170H, and 170I are sent to client 108, along with the new data marker C3 (6332 in FIG.

63B). In one embodiment, a data marker is a synchronization token which is specifically constructed to provide information about the state of information on a client.

In steps 170K and 170L, the client interface module 112 executes these instructions to update the client 108, and saves the new data marker C3 (6318 in FIG. 63B).

Referring back to step 170F, if the server 104 cannot find a previous state of the client 108 corresponding to the data marker contained in the latest sync request from the client 108 (6330 in FIG. 63B), then step 170M is performed. In step 170M, the server 104 identifies the instructions needed to initialize the client 108. In one embodiment, the server 104 initializes the client 108 completely. Control then passes to step 170J, described above.

The full normal synchronization step discussed at step 170E in FIG. 1H1 is shown in FIG. 111. This process applies to a case where the client 108 communicates directly with the server 104.

Control module 142 identifies the deltas in the client databases identified by server 104 during initialization in step 168B (step 172A). In one embodiment of the present invention, a delta is a set of differences between versions of content or, more generally, objects (i.e., different versions of the same pages, documents, links, images, applications, services, etc.). In other words, deltas are sets of differences in the state of the objects currently being offered and the state of the objects in client 108.

Control module 142 sends the deltas to synchronization module(s) 155 via server extension module 156 (step 172B). In an embodiment, these deltas are sent in the synchronization request from client 108 to server 104. This is possible since the client 108 knows which databases the server is interested in. This enables the client 108 to only make one transmission to server 104 during the synchronization process, thereby improving performance.

In one embodiment, synchronization module(s) 155 include web synchronization module 124, fleet management module 154, and/or other

synchronization modules. These modules are responsible for synchronizing to different types of providers 128. Server extension module 156 checks database module 126 to obtain a list of synchronization modules 155 resident on server 104. Depending on the implementation, only some synchronization modules are present on server 104. The server extension module 156 distributes the synchronization responsibilities among the synchronization modules 155. Synchronization modules 155 synchronize the deltas from client 108 with providers 128 (step 172C). Based on the information from provider(s) 128, synchronization modules 155 compile instructions to synchronize the client 108 with providers 128 (step 172D). Synchronization module 155 sends such instructions to client 108, plus updated data marker (step 172E).

Note this is the only transmission from the server 104 to the client 108 during the synchronization process. Thus, the invention achieves a one-up/one-down synchronization process, thereby improving performance. The instructions are transmitted via any reliable transport medium. For example, in one embodiment, HTTP is used. Control module 142 on the client 108 then executes the instructions (step 172F).

FIG. 1I2 illustrates a synchronization process of step 170E (FIG. 1H1) applied to a case where client 108B communicates with server 104 via adapter 118.

Adapter 118 reads data from the client 108 (step 172M). Specifically, adapter interface module 116 reads data from client 108 that includes state information about the resources of the device 106, user specific information, etc.

Adapter 118 identifies deltas in client databases identified by server 104 in step 168B (step 172N). Adapter 118 sends these deltas to synchronization module(s) 155 via server extension module 156 (step 172O). Such deltas are transmitted in the initial synchronization request, thus effecting a "one-up" protocol.

As discussed above, synchronization module(s) 155 on server 104 synchronize deltas from adapter 118 with providers 128 (step 172P).

Synchronization module(s) 155 compile instructions to synchronize client 108 with providers 128 (step 172Q). These instructions are transmitted to the adapter 118, along with the updated data marker (step 172R). This is the only transmission from the server 104 to the adapter 118 during the synchronization process, thus effecting a "one-down" protocol. Adapter 118 then writes the updated data to client 108 (step 172T).

FIG. 1X is another view of the synchronization process. As discussed herein, the device 106 or client 108 provides information about itself and the content it wishes to receive in a single "up" transmission, and the server 104, upon identifying the device 106 or client 108, returns the desired information along with new synchronization changes in a single "down" transmission. Synchronization tokens are passed between client 108 and server 104 so that future transmissions only need to include the information which has changed since the last synchronization session. In other embodiments, a one up and many down synchronization process can be implemented to accommodate the implementation requirements of synchronization modules 155. In still further embodiments, the synchronization session can be implemented on the server 104 by server extension module 156. In such an embodiment, the implementation requirements of synchronization modules 155 would be irrelevant to the "down" transmission, because the server extension module 156 would cache all the information and instructions on behalf of the client and transmit them in all at once.

FIG. 1Y illustrates another view of the synchronization process.

Other synchronization embodiments shall be discussed. It is noted that the synchronization embodiments can be used individually or in combination, as will be appreciated by persons skilled in the relevant art(s).

FIG. 1Z illustrates a granular variable synchronization protocol, according to an embodiment of the present invention. Here, adapter 118 is referred to generically as "PC" and performs the same interface functions as described herein.

FIG. 3A shows flowchart 300 that illustrates the variable granular protocol according to yet another embodiment of the present invention. In step 304, client 108 couples to adapter 118 using a connector or medium (for example, Bluetooth, infrared, etc.).

In step 306, client 108 sends updated information to adapter 118.

In step 308, server 104 receives updated information from adapter 118 (one-up transmission).

In step 310, server 104 examines the updated information and in step 312, server 104 obtains updated information from provider(s) 128.

In step 314, server 104 receives information regarding the sets of content available from provider(s) 128.

In step 316, server 104 constructs a set of content requests for provider(s) 128.

In step 318, server 104 sends requests to provider(s) 128.

In step 320, server 104 receives responses from provider(s) 128.

In step 322, server 104 interacts with client 108 to determine the state of its resources.

As already described herein, client 108 provides state information regarding the nature of its resources. In one embodiment, server 104 assesses the state information preemptively prepared and sent down in order to fit all the required information to the all the necessary device specifications including but not limited to: Dynamic memory specifications, high memory specifications, available storage space, screen size, user profile(s), color depth, applications on device, buttons on-device, data markers, preferences, fonts, sync type, supported data types, supported mime types, and connection/network profile. These types of state information are only for illustration and are not intended to limit the present invention.

In step 324, server 104 optimizes the content received from provider(s) 128. In one embodiment, HTML content is optimized into a tokenized "machine friendly" format which provide specific functionality for client 108. Other

embodiments include optimized formats for XML, JavaScript™, music files, images, etc. For example, as shown in FIG. 1AA, an image is optimized to meet the requirements of client 108 as determined by the state information determined in step 322. As described already herein, image 1A1 is optimized into image 1A2, which may be in color, reduced to fewer colors, etc.

In step 326, server 104 encodes the content received from provider(s) 128 for transmission to adapter 118 and/or client 108. Some of the content may be optimized for display, storage, and/or other functionality on client 108. Some other content may not require any optimization. All content is then encoded for transmission. In one embodiment, the encoding protocol is HTTP. In another embodiment, the transmission protocol is TCP/IP. Various transmissions protocols can be implemented in the present invention with little or no added steps or loss of functionality.

In step 328, adapter 118 signals that it is ready to process content and other network objects from server 104. Client 108 may or may not signal its readiness. In embodiments described herein, client 108 does not communicate with server 104 other than to provide the "up" transmission with all the information required for server 104 to respond completely.

In step 330, server 104 constructs a transmission protocol message for the content and other network objects to be transmitted to adapter 118 or client 108. As discussed with regard to step 326, the transmission protocol selected may determine the characteristics of the message, but not the content of the message.

In step 332, server 104 sends protocol message to providers(s) 128. In one embodiment, the messages sent are queries for forms which were selected and activated by a user of client 108.

In step 334, server 104 receives responses from provider(s) 128 with interactive content. As discussed with regard to step 332, in one embodiment the interactive content can be the responses to form queries.

In step 336, server 104 presents the responses from provider(s) 128 to client 108 and/or adapter 118.

In step 338, server 104 sends modified content to client 108 and/or adapter 118.

In step 340, server 104 sends updated content to client 108 and/or adapter 118.

The variable granularity protocol discussed in FIGS. 1Z, 3A and in the methods of FIGS. 1G - 1I2 do not use file serving or other conventional methods for synchronizing a device to a server or desktop. Instead, the synchronization methods of the present invention can synchronize by using any reliable transport protocol because the delivery of the byte code is transportable in the widest array of delivery protocols. HTTP is one embodiment described herein which is widely implemented and accepted in current computer network topologies. The protocol of the present invention enables operation between client 108 and server 104 and pre-configures the client to preemptively send sets of data to the server 104. This synchronization process of the present invention dynamically checks the need to update or not update content. In one embodiment, it checks the integrity of all data on any page sent via the protocol for the level of granularity. Granularity is determined by a set of deltas. For example, the protocol could acknowledge and read tags associated with HTML and/or XML and sort the objects modeled by theses languages to a client or to a database during a synchronization. In addition this synchronization enables clients with disparate data markers (synchronization tokens) to synchronize by resetting data marker data and maintaining authentication integrity. The features of the present invention described herein are now discussed in more detail with respect to certain embodiments.

As an extension of the variable granular protocol and the use of deltas to determine what should be transmitted up or down, server 104 can deliver XML objects to client 108. The server 104 creates data structures for applications on a client 108 and can receive data structures from the client 108 for conversion to XML for use with a database.

FIG. 3B shows block diagram 350 of one embodiment of the present invention where XML is served to a device 106. Device 106 synchronizes with

server 104 and has its data structure with XML output 354 from database 352. XML table 358 illustrates the format for the results and the structure of inputted queries 356. Layout and rendering module 134 displays the XML output on device 106. This XML output is obtained from server 104 through a XML style and rendering specification 360. Device 106 is able to store and display XML structured information.FIG. 3C shows block diagram 375 of one embodiment of the present invention of off-by-any-number synchronization recovery. See also FIGS. 63A and 63B. FIGS. 1H1 and 1H2 above discuss the steps of this embodiment in detail. FIG. 3C provides additional illustration to aid the explanation and is not intended to limit the present invention.

3.3. Administering Channels, Content, and Data for Mobile Devices

Example administrative related functions are described below. It is noted that these functions are described for illustrative purposes only, and are not limiting.

3.3.1. Cached device information on server

Conventionally, state information on a user or device 106 is stored on the device 106 (such as HTML data markers). Accordingly, functionality to process and maintain such state information resides on a device 106. Locating such functionality on the device 106 may not be optimal in some situations where the resources of the device 106 are limited, such as when the device 106 is a handheld computer.

Accordingly, according to embodiments of the invention, state information (and associated functionality) associated with clients 108 is maintained or cached on the server 104.

FIG. 1J is a flowchart representative of the manner in which state information is cached on server 104.

In step 174A, the client 108 accesses a provider 128 via the server 104.

In step 174B, the provider 128 returns some state information to the server 104. This may be a data marker, for example, or any other type of information on the device/client/user/transaction/etc.

In step 174C, the server 104 maintains such state information on behalf of the client 108. This is performed by the web synchronization module 124 and the database module 126.

In step 174D, the client 108A requests the server 104 to access the same provider 128 as in step 174A.

In step 174E, the server 104 (specifically, the web synchronization module 124) accesses the provider 128 using the state information that is maintained on behalf of the client 108.

3.3.2. Server side optimization of content

When the server 104 obtains an object from a provider 128, the server 104 in some instances passes that object to a client 108. In other cases, however, it may be more efficient for the server 104 to transform the object to a form that is more suitable for use by the client 108. In an embodiment, this transformation is performed by the web synchronization module 124.

This process is represented, for example, in FIG. 1K. Steps 176A - 176C illustrate the initial configuration actions in one embodiment of the present invention. Steps 176D and 176E illustrate an embodiment of any subsequent actions where the client's state information is already stored on the server 104.

In step 176A, client 108 sends state information to server 104 via client communications module 110. State information may contain, among other things, user identity, secure login information, current resources, etc.

In step 176B, server communications module 114 receives client's state information.

In step 176C, server communications module 114 forwards the received state information to database module 126 and web synchronization module 124.

In the case where the client's state information is already stored by database module 126, steps 176D and 176E replace steps 176A - 176C.

In step 176D, client 108 identifies itself to server 104.

In step 176E, server 104 obtains state information about client 108 from database module 126.

In step 176F, the web synchronization module 124 obtains an object from a provider 128. In one embodiment, the object is content which conforms to that which is requested by client's 108 state information, although the object can be any entity, such as an application, service, etc.

In step 176G, the web synchronization module 124 translates/transforms/optimizes the object for use by a particular client. The state information of the device 106 and/or client 108 is considered in this optimization process. The following list of state information is only some of the factors that the web synchronization module 124 considers when performing this optimization (and when determining what, if any, transformations/conversions/optimizations to perform):

Dynamic memory specifications
High memory specifications
Protected memory
Storage memory
Database memory
Available storage space
Screen size
User profile(s)
Color depth
Applications on device
Buttons on-device

Data markers (e.g., cookies, tokens)

Preferences

Fonts

Font specifications

Sync type

Synchronization types

Supported data types

Supported mime types

Connection/Network profile

Other factors will be apparent to persons skilled in the relevant art(s) bases on the teachings contained herein.

FIG. 4A shows block diagram 400 which illustrates the detection 410 of device/client state information by server 104 (or components thereof). In diagram 400, the synchronization process 412 includes only kinds of content 414 supported by various devices/clients 416.

3.3.3. Hashed device state

In embodiments, during synchronization operations, prior to sending an object to a client 108, the server 104 checks to see if the object differs from the instance of the object already resident on the client 108. If the object is the same as that already resident on the client 108, then the server 104 does not send the object to the client 108. This process is illustrated in FIG. 1L.

In step 178A, the client 108 requests an object (directly or indirectly).

In step 178B, the web synchronization module 124 obtains the requested object from a provider 128.

In step 178C, the web synchronization module 124 performs a hash operation on the object and compares the hash result to a previously stored hash result for the object.

In step 178D, the web synchronization module 124 determines if the hash result generated in step 178C is the same as the previously generated and stored hash result of the object.

In step 178E, if they are the same, then the client 108 is informed that the object has not changed.

In step 178F, if they are not the same, then object is transformed as in step 176B of Fig. 1K. Also, the new hash value generated in step 178C is stored by the server 104.

In step 178G, the web synchronization module 124 performs a hash operation on the transformed object.

In step 178H, the web synchronization module 124 compares the hash result of the transformed object to a previously stored hash result of the transformed object.

In step 178I, the web synchronization module 124 determines if the hash result generated in step 178G is the same as the previously generated and stored hash result of the transformed object.

In step 178E, if they are the same, then the client 108 is informed that the object has not changed.

In step 178J, if they are not the same, then the transformed object is sent to the client 108. Also, the new hash value generated in step 178G is stored by the server 104.

Thus, according to embodiments of the invention, the server 104 determines whether current versions of objects already reside on clients 108 by using hash results, as opposed to the objects themselves. This reduces the amount of memory needed on the server 104 (since only the hash results need to be stored, not the objects themselves).

Also, according to embodiments, there are two checks to see if the current versions of objects already reside on clients 108. The first check is made to the raw object, and the second check is made to the transformed versions of the objects.

FIG. 4B shows block diagram 402 illustrating the hashed device state process described herein. In diagram 402, Server 104 synchronizes with device 106 identifiers for updated device information 422. The synchronization 422 includes state information 426 and data 427 that is stored by server 104 using global unique identifiers (GUIDs) 424 for each object. GUIDs provide the hash object for the hash operations/comparisons described herein. A hash value is the result of a hash operation. In an embodiment, a hash value is a numerical fingerprint of any amount of data. In one embodiment, hash values are calculated for each HTML document. This hash value is smaller and more efficient to store on server 104. Additionally, server 104 can compare two or more hash values more readily and faster than comparing the complete documents than the hash values. It is noted that in embodiments, data 427 have hash operations performed on them more frequently than state information 426. For example, the screen size of a device will likely remain constant while the data on the device changes repeatedly.

3.3.4. Syncing music, movies, books, photo albums, and other collections of objects

The invention supports channels which comprise web sites having collections of objects, such as collections of music, images, books, movies, applications, services, etc. By selecting such a channel, the client 108 can be populated with such collections of objects.

For example, if a channel having a collection of music is selected, then it is possible to turn the client 108 into a "jukebox" once the music collection is stored on the client 108 during the synchronization process. Similarly, a client 108 can become a photo album, a book library, a movie theater, an application library, etc., by selecting appropriate channels. This process is represented by FIG. 1M. It is noted that this process is applied to collections of music, but it is also applicable to collections of any types of objects. It is also noted that a given

channel may have combinations of different types of objects, such as combinations of music, movies, applications, images, services, etc.

In step 180A, a provider 128 is added to list of the channels supported by the server 104. The provider 128 offers a collection of objects.

In step 180B, a user of client 108 selects the channel 128.

In step 180C, during the next synchronization operation, the selected channel is synchronized with client 108.

3.3.5. Fleet Management

The invention supports organizing groups of clients 108 as "fleets." For example, all clients 108 associated with employees of a company, or of a department of a company, may be a fleet. As another example, client 108 in a family can be a fleet. Generally, any group of clients 108 can be a fleet.

The invention supports performing mass operations on or relating to clients 108 in a given fleet (or multiple fleets). This process is shown, for example, in FIG. 1N.

In step 182A, a desired mass operation is defined. For example, one may define a mass operation to be the collection and processing of state information relating to clients 108 in a fleet. Another operation could involve installing an application on all clients 108 in a fleet. In embodiments, a third party is permitted to define the mass operation by paying some amount to the server 104 (specifically, by providing some compensation or consideration to the entity associated with or responsible for the server 104).

In step 182B, the fleet or fleets are identified.

Steps 182D and 182C/182E illustrate processing relating to two types of mass operations.

In step 182C, cached information in server 104 relating to the clients 108 in the identified fleet(s) are collected and processed in a manner defined by the mass operation defined in step 182A. Optionally in step 182E, perhaps upon

payment by a third party of some consideration to the entity associated with or responsible for server 104, such information is used in a manner desirable to the third party (such as providing marketing information to the third party).

In step 182D, during synchronization with the clients 108 in the fleet(s), the desired mass operation defined in step 182A is performed on the client 108 (such as upgrading software on the clients 108).

3.4. Customizing Channels, Content, and Data

3.4.1. Creating Custom Channels

As described herein, the invention provides a user or an administrator with the ability to navigate or "surf" to a Web page or site and automatically add that Web page or site to the user's list of channels and/or to the channels provided by server 104, respectively. FIG. 1O is a flow diagram describing a process 184 for automatically adding a Web page or Web site to a user's list of channels from a Web browser. Although flow diagram 184 is described from a user's perspective, flow diagram 184 is equally applicable to an administrator for automatically adding channels provided by server 104. Process 184, as well as other applicable processes described herein, may be performed using a desktop Web browser, such as Internet Explorer, developed by Microsoft and Netscape Communicator, developed by Netscape, or other browsers. Process 184 begins with step 184A.

In step 184A, the user bookmarks an automatic channel link from the Web page of server 104. Bookmarking identifies the URL (Uniform Resource Locator) of the automatic channel Web page. For example, with Internet Explorer 4.0, the user creates a bookmark by right clicking on the automatic channel link and selecting "Add to Favorites." In another example, using Netscape Communicator 4.0, a bookmark is created by right clicking on the automatic channel link and selecting "Add Bookmark." Bookmarking is a well

known technique to those skilled in the relevant art(s). The process proceeds to step 184B.

In step 184B, the user surfs to a Web site. The user determines that he/she wishes to create a channel. In step 184C, the user invokes the automatic channel bookmark. The process then proceeds to step 184D, where the user is brought back to the automatic channel Web page. The URL of the preceding Web site is now available to server 104. The process proceeds to step 184E.

In step 184E, in an embodiment, a script in the automatic channel Web page queries the browser for the URL of the previous Web site and automatically populates an automatic channel form for the Web site to be added to the user's list of channels. In another embodiment, a header is used to determine the URL of the previous Web site. The automatic channel form contains fields identifying the title and URL of the Web site, the maximum channel size, the link depth, whether images are to be included, whether to follow off-site links, when to refresh, etc. The process proceeds to step 184F.

In step 184F, the user reviews the channel settings that were automatically generated, and, if satisfied, selects the save channel button to save the channel settings. If the user is not satisfied, the user may modify the settings to the user's satisfaction and then select the save button to save the channel settings. The process proceeds to step 184G.

In step 184G, user interface 130 causes a new channel to be added to the user's list of channels. The new channel is entered in database module 126 of server 104. On the next sync of client 106, the new channel will be synced to the client.

The invention also allows a provider 128 to enable a user of device 106 to have the Web page of provider 128 loaded on the user's device 106. Provider 128 provides a link or quick channel button on its Web page that, if selected by the user, indicates that the user would want the Web page converted into a channel that is loaded on his/her mobile device. This process does not require that the user be a registered user of server 104. FIG. 1S describes a process 192

for allowing a provider 128 to enable a user of device 106 to have the Web page of provider 128 converted into a channel that is then loaded on device 106. As described herein, device 106 may be a mobile device. Process 192 begins with step 192A.

In step 192A, a user surfs to the Web page of a provider 128. Provider 128 has a quick channel button that, when selected by the user, links the user to a Web page of server 104. The process proceeds to step 192B.

In step 192B, the user selects the quick channel button indicating that the user would like the Web page converted into a channel that is loaded on his/her device 106. In step 192C, the user is automatically sent to the Web site of server 104 by selecting the quick channel button. The process proceeds to step 192D.

In step 192D, the server attempts to determine whether the user is a registered user of server 104. The process proceeds to step 192E.

In step 192E, server 104 determines if the user is a registered user of the Web site of server 104. If the user is detected to be a registered user with the Web site of server 104, the process proceeds to step 192F. If the user cannot be detected as a registered user of the Web site of server 104, the process proceeds to step 192G.

In step 192G, server 104 queries the user as to whether or not the user is registered with the server 104. If the user's response is yes, the process proceeds to step 192I. In step 192I, the user logs on to server 104. The process then proceeds to step 192F.

Returning to step 192G, if the user response is no, that he/she is not registered as a user of the Web site of server 104, the process proceeds to step 192H. In step 192H, the server allows the user to register by taking the user through the registration process. The process proceeds to step 192F.

In step 192F, server 104 adds a new channel (that is, the Web site of the content provider having the quick channel button) to the user's list of channels, returns the user to the Web site from which it was linked, and presets state

information to enable the Web site to be displayed on the user's device 106 on the next syncing operation.

3.4.2. Tags

As described herein, the invention delivers Web content to clients 108. Server 104 optimizes the Web content display to enable the display to fit within the parameters of the client 108. Such parameters may include, but are not limited to, dynamic memory specifications, high memory specifications, protected memory, storage memory, database memory, available storage space, screen size, user profile(s), color depth, applications on the device, buttons on the device, data markers, preferences, fonts, font specifications, sync type, supported data types, supported mime types, and connection/network profile. FIG. 1AA illustrates an optimization of a Web site page for display on a handheld device. As shown in FIG. 1AA, Web page graphic display 1A1 is from a large screen desktop display and Web page graphic display 1A2 is an optimized version 1A2 of Web page graphic display 1A1 that has been optimized to fit on a handheld device, such as device 106.

The invention also identifies Web content that is designed for additional modifications. Server 104 identifies the additional modifications through the use of tags. Any and all bytes processed by server 104 are potentially examined for compression. Server 104 detects the tag and executes the necessary logic. FIG. 1P is a flow diagram describing an overview of a process 186 for handling predefined tags by server 104 and clients 108. The process begins with step 186A.

In step 186A, providers 128 create Web page content using predefined tags to optimize use on devices 106. The process proceeds to step 186B.

In step 186B, server 104 and clients 108 process objects within the Web page using tags contained therein.

FIG. 1Q is a flow diagram describing process 186B in greater detail when a client 106 accesses a Web page having predefined tags. The process begins with step 188A.

In step 188A, client 108 accesses a Web page cached in client 108 or provided to client 108 by server 104.. The process proceeds to step 188B.

In step 188B, client 108 processes the Web page based on the presence or absence of tags. For example, META tags may be used. An example META tag is <META NAME = "Handheld-Friendly" content = "True">. This META tag (if set to true) enables several HTML features that are normally turned off. For example, most TABLES, HSPACEs, and VSPACES are designed for much larger screens, and are therefore not usually processed by client 108. However, TABLE tags are displayed, and HSPACE and VSPACE attributes of IMG (image) tags are processed if the page is marked as "Handheld Friendly." Another exemplary tag is an <AGIGNORE> or </AGIGNORE> tag used in a wireless channel. The AGIGNORE tag is used to surround content within an HTML page that may be inappropriate or unattractive on Internet-based phones. Content surrounded by this tag is ignored by client 108. A tag also exists to control how JavaScript™ is handled. If the tag exists, JavaScript™ behavior is enabled. If the tag does not exist, JavaScript™ behavior is ignored.

A page tracking tag may be used that enables client 108 to report to server 104 the number of times a user has viewed a Web page (in embodiments, the client 108 may report other client activity). A page break tag, <PAGEBREAK TITLE = "your title"> is used in a wireless channel. Such a tag breaks up pages on request. When processing pages for devices other than WAP (Wireless Application Protocol) phones, server 104 ignores the page break tag.

FIG. 1R is a flow diagram describing process 186B in greater detail when a server 104 accesses a Web page having predefined tags. The process begins with step 190A.

In step 190A, server 104 accesses a Web page. The process proceeds to step 190B.

In step 190B, server 104 processes the Web page based on the presence or absence of tags, as previously described above. In one embodiment, tags exist for server 104 that indicate whether to follow a link, not to follow a link, or to follow a link a number of layers (N) down.

The invention is not limited to the tags described above. One skilled in the relevant art(s) would know that other types of tags may be used without departing from the scope of the present invention, based on the teachings contained herein.

3.4.3. Client Registration Process Embodiment

In one embodiment, the invention includes a client registration process that includes GUI elements for the capture and configuration of client details and preferences. The invention is not limited to all of the steps described herein. One skilled in the relevant art(s) would know that other steps may be used without departing from the scope of the present invention, based on the teachings contained herein.

In one embodiment, the invention registers a user from the Web site of server 104. FIGS. 5A through 5J are flow diagrams describing the registration process. In FIG. 5A, the process begins with step 502. In step 502, a user arrives at the home page of server 104. The user may arrive at the home page from a link or a direct URL. If the user has previously registered, the user is identified via a data marker, and the user's identification is displayed on the screen.

In step 504, if the user is a current user, the process proceeds to step 508. If the user is not a current user, the process proceeds to step 506.

In step 506, a new user is registered. The registration process for a new user is described below with reference to FIGS. 5I -5J.

In step 508, the current user may select between an editing option for editing their account or an add channel(s) option for adding additional channels

to their account. If the current user selects the add channel(s) option, the process proceeds to step 512.

In step 512, the user may add new channels using an "Add featured channels" option or the user may add new channels via an explore and add new channels option. If the user adds new channels via the "Add featured channels" option, the process proceeds to step 514.

In step 514, the user may select channels from a featured channels list. The added channel is displayed in a smart little unit (SLU) labeled "user account." An indicator will appear beside the added channel to indicate that the channel has not been synchronized with client 108 (step 516).

Returning to step 512, if the user adds new channels via the "explore and add new channels" option, the process proceeds to step 518. In step 518, the user selects a category from a directory of channel categories. A list of Web sites from the selected category is displayed in step 520. In step 522, the user may explore any Web site in the list and/or add any of the Web sites to their account. If the user adds a Web site to their account, the process proceeds to step 516.

Returning back to step 508, if the user selects the edit account option, the process proceeds to step 510. In step 510, the user is linked to a "user channel" page of the Web site of server 104. The process then proceeds to step 524 in FIG. 5B.

In step 524, the user is presented with a plurality of options. The user may add and remove channels, export channels, import older channels and data files for viewing, alter settings, upgrade the software, or view their sync history. If the user wishes to view their sync history, the process proceeds to step 526.

In step 526, the user selects the sync log option. The user's sync history is accessed and displayed in step 528.

Returning to step 524, if the user wishes to alter their account settings, the process proceeds to step 530. In step 530, the user selects the settings option. The user is then linked to an account settings page in step 532. In step 534, the

user is queried to change their password and email address, as well as other user information.

Returning back to step 524, if the user wishes to export channels, the process proceeds to step 538. In step 538, the user selects the export channels option. URLs from the user's channels are automatically generated for adding the channels to another user account.

Returning back to step 524, if the user wishes to import older channels and data files, the process proceeds to step 585. In step 585, the user selects the import option. Old versions of channels and data files are then selected and displayed in step 587.

Returning back to step 524, if the user wishes to modify their existing channel(s), the process proceeds to step 581. In step 581, the user clicks on a channel to edit. Once the clicks on the channel, the channels parameter settings are displayed. In step 583, the user may edit the channel parameter settings. Channel parameter settings may include, but are not limited to, the channel name, root URL, an images option, link depth, an offsite links option, maximum size, and refresh period.

Returning back to step 524, if the user wishes to add a customized channel, the process proceeds to step 540 in FIG. 5C. In step 540, the user selects a "create custom channel" option. The user is then queried to manually enter information for adding a user favorite Web site to the SLU. The process then proceeds to step 516 in FIG. 5A, where the added channel is entered into the SLU.

Returning back to step 524 in FIG. 5B, if the user wishes to add a channel automatically, the process proceeds to step 544 in FIG. 5C. In step 544, the user selects the automatic channel option. A channel is then created while surfing the Web when the user clicks a button or chooses a bookmark, as described with reference to FIGS. 1O and 1S. The process then proceeds to step 516 in FIG. 5A, where the added channel is entered into the SLU.

Returning back to step 524 in FIG. 5B, if the user wishes to remove a channel, the process proceeds to step 548 in FIG. 5C. In step 548, the user removes a channel by checking the channel to be deleted in the SLU and selecting the delete button.

Returning back to step 524 in FIG. 5B, if the user wishes to update the software, the process proceeds to step 550 in FIG. 5D. In step 550, the user selects the software setup option. The user is then linked to a software setup page in step 552. The process proceeds to step 554.

In step 554, the user may either download the software or configure a client communication module 110. If the user selects the configure client communication module 110, the process proceeds to step 562.

In step 562, the user is linked to a configure client communications page. In step 564, the user must click the select button to begin the configuration. Therefore, the client communication module 110 is configured to identify the particular server that will send data to device 106.

In step 566, the software is configured to communicate with the appropriate servers. The process then proceeds to step 590 in FIG. 5H.

In step 590, a browser prompts the user to sync the device. In one embodiment, a browser prompts the user to place their device in a cradle and synchronize the device. In step 592, the user synchronizes the device. A notification from the browser will be displayed to indicate that the client communication module 110 process is complete.

Returning to step 554 in FIG. 5D, if the user wishes to download the software, the download software option is selected. The process proceeds to step 556.

In step 556, the software is downloaded. In step 558, a notification is displayed to indicate that client 108 will be installed on device 106 upon synchronization.

In step 560, the browser is displayed to prompt the user to sync the device. In one embodiment, the browser prompts the user to place the device in

the cradle and press the sync button to sync the device 106. The process then proceeds to step 568 in FIG. 5E.

In step 568, the browser queries the user as to whether the user wants to use their present account or create a new account. In step 570, if the user selects to use their existing account, the process proceeds to step 562 in FIG. 5D to configure the client communication module 110. If, in step 570, the user selects to create a new account, the process proceeds to step 572. In step 572, a new account for the user is created. The process then proceeds to step 562 in FIG. 5D to configure the client communication module 110.

The process of configuring the software to communicate with the servers, such as server 104, will now be described in greater detail in FIG. 5F. The process begins with step 574.

In step 574, it is determined whether the user is a pre-existing user. If the user is a pre-existing user, the user is queried for the type of device that will use the server in step 576. In step 578, the user selects the type of device. The process then proceeds to step 580.

Returning to step 574, if the user is not a pre-existing user, the process proceeds to step 580.

In step 580, details of the configuration are displayed. The process proceeds to step 582.

In step 582, the user may replace their existing server profile or add another server profile. Many reasons may exist as to why a single user may want different server profiles. For example, a first server profile may contain only sports channels and a second server profile may be directed to stock channels. The process then proceeds to step 584 in FIG. 5G.

The user has the option of testing their settings. In step 584, the user is queried as to whether the user desires to test their settings. If the user selects the test settings option, the process proceeds to step 586. In step 586, the results of the tests are displayed. The process then proceeds to step 588.

Returning to step 584, if the user does not select the test settings option, the process proceeds to step 588.

In step 588, the process ends.

FIG. 5I is a flow diagram illustrating registration process 506 for a new user. The process begins with step 501. In step 501, the user selects the sign-up prompt. The user is then linked to the software setup page in step 503.

In step 505, the user selects the type of mobile device the user will be synchronizing.

In step 507, a security notification pertaining to import/export laws is displayed.

In step 509, the user selects the download software prompt. The software is downloaded in step 511. A notification is displayed indicating that the software has been successfully downloaded in step 513.

In step 515, the browser prompts the user to sync the device. In one embodiment, the browser prompts the user to place their device 106 into a cradle and initialize the sync process for the device. The process then proceeds to step 517 in FIG. 5J.

In step 517, the user is prompted to enter their user account information. The user account information may include, but is not limited to, the user's name, address, email address, and password. The process then proceeds to step 519.

In step 519, the browser is displayed for configuring the client communication module 110. The client communication module 110 application must be informed of the server that will send data to the user's device 106. The client communication module 110 is processed in a similar matter as described in steps 562, 564, and 566 of FIG. 5D.

In step 521, it is determined whether the new user is pretending to be a new user or is a real new user. If the user is pretending to be a new user, the process proceeds to step 523, where steps 582 - 588 in FIGs. 5F and 5G are performed to allow the user to replace their existing server profile, and to test the settings, if desired. If the user is an actual new user, the process proceeds to step

525. In step 525, the user is instructed to perform the synchronization process as described in steps 590-594 in FIG. 5H.

In one embodiment, a user may have access to a user account without having to be registered. FIG. 1T is a flow diagram describing a process for enabling an unregistered user to access server 104. The process begins with step 194A.

In step 194A an unregistered user invokes client 108. Control then proceeds to step 194B.

In step 194B, client 108 is synchronized with server 104. Server 104 assigns client 108 an anonymous account in step 194C. The process proceeds to step 194D.

In step 194D, server 104 periodically reminds client 108 to register. It is not a requirement that the user register. If the user registers, in step 194E, the user is promoted to a registered user.

3.4.4. Selecting and Organizing Channels for a Client

As described herein, the invention allows a user to select and organize channels for client 108 of device 106. FIG. 5K is a flow diagram describing a method for selecting and organizing channels. A process 531 begins with step 533. In step 533, a SLU, also referred to as a basket or cart, displays the current channels in a user's account. The SLU is representative of the user's account. In step 535, the SLU enables the user to create, remove, or modify channels.

FIG. 5L is a flow diagram describing in greater detail, process 535 for creating, modifying, and removing channels. The process begins with decision step 537. In step 537, if the user wishes to create channels to place into their SLU, the process proceeds to step 539.

In step 539, the user may select the create channel button displayed in the SLU. In step 541, the user is then linked to the create channel page, where the user may manually enter information to add a favorite site to their account. The

user may also create a new channel automatically using the automatic channel methods described above.

Returning to decision step 537, if the user wishes to remove channels from their SLU, the process proceeds to step 543. In step 543, the user selects the removal link associated with the channel to be deleted in the SLU. In step 545, the channel is deleted.

Returning to decision step 537, if the user wishes to modify channels that are contained in their SLU, the process proceeds to step 547.

In step 547, the user selects the edit button displayed on their SLU. The edit button links the user to the "user channels" page. In step 549, the user may click on a channel to be modified. Parameter data for that channel is then displayed, the parameter data may include, but is not limited to, the channel name, root URL, maximum size, link depth, an offsite links option, and refresh period setting.

In step 551, the user may modify the parameter(s) for the channel.

Device 106 may contain a channel manager. The channel manager operates in a similar manner as the SLU.

3.4.5. An Account Management Process

The invention also allows a user to manage their accounts without having to use the registration process for a current user. A flow diagram describing an account management process is shown in FIG. 5M. With the account management process, a user may choose to alter the settings of their account, update the software, view their sync history, create custom channels manually or automatically, export channels, import channels, modify channels, and delete channels. The process begins with step 555. In step 555, a user is allowed to manage their account from the "user channels" page of the Web site for server 104.

If the user wishes to alter their settings, the process proceeds to step 557. Account settings, such as the user's password and/or email address, may be altered in a similar manner as described in FIG. 5B.

If the user wishes to update the software, the process proceeds to step 563. The software may then be updated in a similar manner as described in FIGS. 5D, 5E, and 5H.

If the user wishes to view their sync history, the process proceeds to step 565. The user may view their sync history in a similar manner as described in FIG. 5B.

If the user wishes to create customized channels manually, the process proceeds to step 567. Customized channels are created in a similar manner as described in FIG. 5C.

If the user wishes to create channels automatically while surfing, the process proceeds to step 569. Channels are automatically created in a similar manner as described in FIG. 5C.

If the user wishes to export channels to other users, the process proceeds to step 571. The user's channels are exported to other users in a similar manner as described in FIG. 5B.

If the user wishes to import old channels and data files, the process proceeds to step 573. Old channels and data files are imported for viewing as described in FIG. 5B.

If the user wishes to modify channels, the process proceeds to step 575. Current user channels may be modified in a similar manner as described in FIG. 5B.

If the user wishes to remove channels from their account, the process proceeds to step 577. Channels are removed from the user's account in a similar manner as described in FIG. 5C.

4. Example User Interface Screen Shots

Various example screen shots related to the functionality of the invention are considered in this section. It is noted that these screen shots are provided for illustrative purposes only, and are not limiting. Additional screen shots will be apparent to persons skilled in the relevant art(s).

These screen shots are generated by the user interfaces of the invention, such as user interface 130 in the server 104 and user interface 144 in the clients 108. However, other modules of the invention may also contribute to the user interface function with regard to their respective functionalities and responsibilities. For example, and without limitation, the forms module 136 may contribute to user interface functionality with regard to forms.

Generally, screen shots are generated to enable interaction with users. For example, screen shots may be generated to provide information to users, or to obtain information from users. Other uses of screen shots will be apparent to persons skilled in the relevant art(s).

The screen shots in FIGS. 6-62 depict functionality of embodiments of the invention. The invention is directed to such functionality.

FIG. 6 is an example screen shot generated by forms module 136. This screen shot shows the status of forms completed by the client 108. Via this screen, the client 108 may obtain additional information on forms, and may manipulate the forms (such as delete selected forms).

FIG. 7 is a screen shot of an example completed form displayed on a client 108.

FIG. 8 is an example screen shot relating to the channel manager displayed on a client 108 (preferably, but not limited to, a device operating according to Windows CE). In this screen, the client 108 can remove channels, as well as perform other administrative tasks on channels.

FIG. 9 is an example screen shot displayed on a client 108 relating to browsing options on client 108.

According to the invention, the client 108 can cache web pages in the databases of the client 108 when it is browsing the Internet (while connected to

the server 104, for example). Then, when not connected to the Internet, the user of the client 108 may browse and interact with pages stored in the cache. FIG. 10 is an example screen shot that enables the client 108 to delete pages from the cache.

FIG. 11 is an example screen shot of an application menu displayed on the client 108. Item 1102 enables the user of the client 108 to access the functions of the client 108.

FIG. 12 is an example screen shot representing a channel subscription page displayed on the client 108. When in the off-line mode, a user of the client 108 can elect to subscribe to channels listed in the channel subscription page. In an embodiment, the selected channels are loaded on the client 108 during the next synchronization operation.

FIG. 13 is an example screen shot of a find function available on the client 108.

According to an embodiment of an invention, a corporate entity (or enterprise) controls a server 104, and its employees have devices 106 that interact with the server 104 in the manner discussed herein. The server 104 may support channels that are specific to the enterprise, or otherwise relevant to the enterprise (as well as supporting any other channels). FIG. 14 is an example screen shot of a home page for an enterprise having a server 104. The home page includes personal channels 1402 and group channels 1404. FIG. 20 shows an example screen shot corresponding to an enterprise specific channel that is displayed on the client 108.

FIG. 15 is an example screen shot of a home page that is displayed on the client 108 when the client 108 connects to the server 104. The client 108 may be connected to the server 104 via a wireless link, for example (although the invention is not limited to this example).

FIG. 16 is an example notification message that is displayed on the client 108 when the client 108 attempts to access a web page or other object that is not

resident on the client 108 (either because the object was not loaded on the client 108 during the sync process, or the page is not in the on-device cache).

FIG. 17 is an example screen shot displayed on the client 108 showing preferences for operation on the client 108 (preferably, but not limited to, devices 106 using the Palm operating system).

FIG. 18 is an example screen shot displayed on the client 108 wherein a user may enter a URL to retrieve an object corresponding to the URL. When connected to server 104, server 104 will retrieve the object at the URL (although an attempt is first made to locate the object on the client 108). When not connected to the server 104, the on-device cache is checked for the object. If the object is not found, then in an embodiment the request for the URL is cached and then processed during the next sync (an example notification screen is shown in FIG. 19).

FIG. 19 is the confirmation message for action 162D of FIG. 1D.

FIG. 20 is a sample enterprise application optimized for use on the client in a mobile device.

FIGS. 21 and 22 indicate that it is possible to change fonts and font sizes on the client 108 to enhance display quality.

FIG. 23 shows an example screen shot displayed on a client 108 showing example navigation controls: links 2302, back 2304, forward 2306, home 2308, and scroll 2310.

FIG. 24 shows an alternative menu/tool bar displayed on some clients 108 (such as clients 108 operating according to Windows CE environment). FIG. 25 illustrates a home page and FIG. 26 illustrates a find function displayed on some clients 108 (such as clients 108 operating according to Windows CE environment). FIG. 27 illustrates an example home page for an enterprise displayed on some clients 108 (such as clients 108 operating according to Windows CE environment).

FIG. 28 shows an example enterprise server home page.

FIG. 29 shows example enterprise user interface naming conventions.

FIG. 30 illustrates an example enterprise software architecture.

FIG. 31 shows an example screen shot to enter new channels.

FIG. 32 shows an example screen shot to set refresh properties. It is noted that the user of the client 108 is permitted to indicate whether a particular object is refreshed at each sync, only once per day, or according to some other schedule. Accordingly, when a statement is made herein that some channel, object or other entity is loaded on the client 108 "during the next sync" (or using similar language), it should be understood that loading of the object/entity on the client 108 may occur during some other future sync operation (not just at the "next sync").

FIG. 33 shows an example screen shot to update user membership in a group.

FIG. 34 shows an example edit server profile dialog.

FIG. 35 shows an example screen shot to modify channels, and to display a sync history for a client 108.

FIG. 36 shows an alternative view of the architecture of embodiments of the invention.

FIG. 37 shows an example enterprise server status page.

FIG. 38 shows an example find user page.

FIG. 39 shows an example user detail/account information page.

FIG. 40 shows an example screen shot to modify groups.

FIG. 41 shows an example screen shot showing group information.

FIG. 42 shows an example screen shot to change admin passwords.

FIG. 43 shows an example screen shot to establish channel properties when adding a channel to the collection of channels supported by server 104.

FIG. 44 shows an example screen shot regarding a process for automatically adding channels, as described elsewhere herein.

FIG. 45 shows an example screen shot to create a channel to add to the collection of channels supported by server 104.

FIG. 46 shows an example personal account page. This page could be displayed on the device 108 or on some computer connected to the server 104 through any means, such as but not limited to the Internet. This page shows a list 4602 of the channels to which the user (or client 108) is currently subscribed. The page also shows a list 4604 of featured channels. The channels in the list 4604 of featured channels can be selected according to any procedure. For example, this list 4604 may include most frequently selected channels. Also, the list 4604 may be compiled based on financial considerations. For example, providers 128 may pay some compensation to have their channels in the featured channels list 4604. This compensation may be adjusted based on which slot on the featured channels list 4604 they have (such as the top spot 4606, the second spot 4608, etc.).

FIG. 47 shows an example channel manager page. This page could be displayed on the device 108 or on some computer connected to the server 104 through any means, such as but not limited to the Internet. In this page, the user can delete channels to which he is currently subscribed. The user can also add channels (to his or other users' accounts) via options designated as 4702. An icon is displayed if an account needs to be synced with the client 108.

FIG. 48 shows an account settings page to enable a user/client 108 to make changes to his account. This page could be displayed on the device 108 or on some computer connected to the server 104 through any means, such as but not limited to the Internet.

FIGS. 49-62 relate to registering a new client 108. These pages could be displayed any data processing device connected to the server 104 through any means, such as but not limited to the Internet.

FIG. 49 shows an example software set up page. In this page, the user can elect which version of client software to download. FIG. 50 shows a display box that indicates the state of downloading the software selected in FIG. 49. After the software is downloaded, a notifier box shown in FIG. 51 informs the user that the client software will be install on the device 106 during the next sync

between the device 106 and the computer that contains the software downloaded via FIGS. 49-50. FIG. 52 is an instruction screen to help the user load the client software on device 106.

After the client software is installed on device 106, example screen shot in FIG. 53 is displayed. In this page, the user is able to indicate whether he is an existing user and wishes to use his existing account, or whether he wishes to open a new account. If the user indicates that he wants to open a new account, then the registration process follows (described elsewhere herein).

FIG. 54 is an example screen shot to enable the client 108 to configure client software to enable the client 108A to communication with the server 104. In an enterprise environment, clicking button 5402 will configure the client 108A to speak with the enterprise server 104 (i.e., a typically private server 104 controlled by an enterprise). In a non-enterprise environment, clicking button 5402 will configure the client 108A to speak with a non-enterprise server 104 (i.e., a typically publicly available server 104).

FIG. 55 is an example page that enables the user to select which device 106 to configure for communication with the server 104 (the user may have multiple devices 106).

FIG. 56 is an example summary page indicating the user's selections from FIGS. 54 and 55.

FIG. 57 is an example page that enables the user to indicate whether he wishes to add another server profile, or to overwrite the existing server profile.

FIG. 58 is an example page that enables the user to test the connection between client 108A and server 104. FIG. 59 is an example page displayed if this test is successful. FIG. 60 is a summary page that indicates the state of the status of the set up process.

FIG. 61 is an example page that provides information regarding syncing with the server 104.

FIG. 62 is an example page confirming successful registration with server 104.

5. *Conclusion*

While various embodiments of the present invention have been described above, it should be understood that they have been presented by way of example only, and not limitation. It will be understood by those skilled in the art that various changes in form and details may be made therein without departing from the spirit and scope of the invention as defined in the appended claims. Thus, the breadth and scope of the present invention should not be limited by any of the above-described exemplary embodiments, but should be defined only in accordance with the following claims and their equivalents.

What Is Claimed Is:

1. A method for enabling a user to place network content on a client, comprising the steps of:
 1) requesting to view content on said client;
 2) interacting with a server to identify one or more channels;
 3) synchronizing said client with said server; and
 4) gathering said one or more identified channels to be sent to said client;
 5) caching said one or more channels on said client.

2. The method of claim 1, further comprising the step of surfing said one or more channels off-line by reading one or more cached pages.

3. A method for synchronizing a client with a server, comprising the steps of:
 1) receiving a synchronization query from said client, wherein said synchronization query includes a current data marker;
 2) using said current data marker to determine whether a previous synchronization process was successful; and
 3) performing a full synchronization process if said previous synchronization process was successful.

4. A method for enabling a user to create customized channels for mobile devices, comprising the steps of:
 1) creating a bookmark to an automatic channel Web page;
 2) surfing to a Web site;
 3) invoking said automatic channel bookmark, thereby displaying said automatic channel Web page;

4) querying a browser via said automatic channel Web page for a URL (Universal Resource Locator) of said Web site;

5) automatically populating an automatic channel form for said Web site; and

6) automatically adding said Web site as a new channel to be stored in a database.

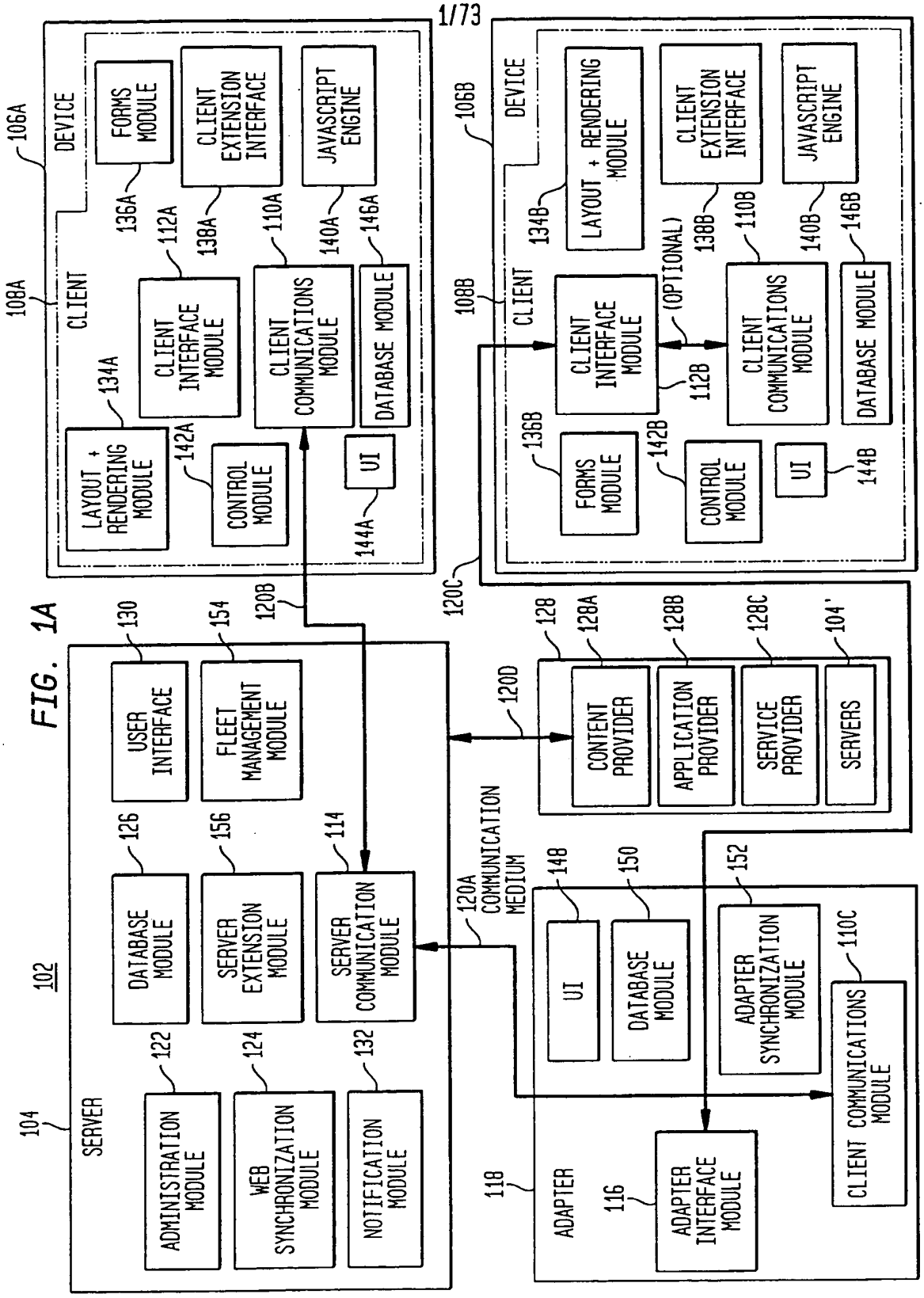

106A
DEVICE
108A
CLIENT
FORMS MODULE
136A
CLIENT EXTENSION INTERFACE
138A
JAVASCRIPT ENGINE
140A
112A
110A
146A
CLIENT INTERFACE MODULE
CLIENT COMMUNICATIONS MODULE
DATABASE MODULE
134A
LAYOUT + RENDERING MODULE
142A
CONTROL MODULE
UI
144A
106B
DEVICE
108B
CLIENT
LAYOUT + RENDERING MODULE
134B
CLIENT EXTENSION INTERFACE
138B
JAVASCRIPT ENGINE
140B
110B
146B
CLIENT INTERFACE MODULE
(OPTIONAL)
CLIENT COMMUNICATIONS MODULE
DATABASE MODULE
112B
FORMS MODULE
136B
CONTROL MODULE
142B
UI
144B
120C
120B
130
USER INTERFACE
154
FLEET MANAGEMENT MODULE
120D
128
128A
128B
128C
104'
CONTENT PROVIDER
APPLICATION PROVIDER
SERVICE PROVIDER
SERVERS
102
126
DATABASE MODULE
156
SERVER EXTENSION MODULE
114
SERVER COMMUNICATION MODULE
120A
COMMUNICATION MEDIUM
148
UI
150
DATABASE MODULE
152
ADAPTER SYNCHRONIZATION MODULE
110C
CLIENT COMMUNICATIONS MODULE
104
SERVER
122
ADMINISTRATION MODULE
124
WEB SYNCHRONIZATION MODULE
132
NOTIFICATION MODULE
118
ADAPTER
116
ADAPTER INTERFACE MODULE

FIG. 1A

FIG. 1B

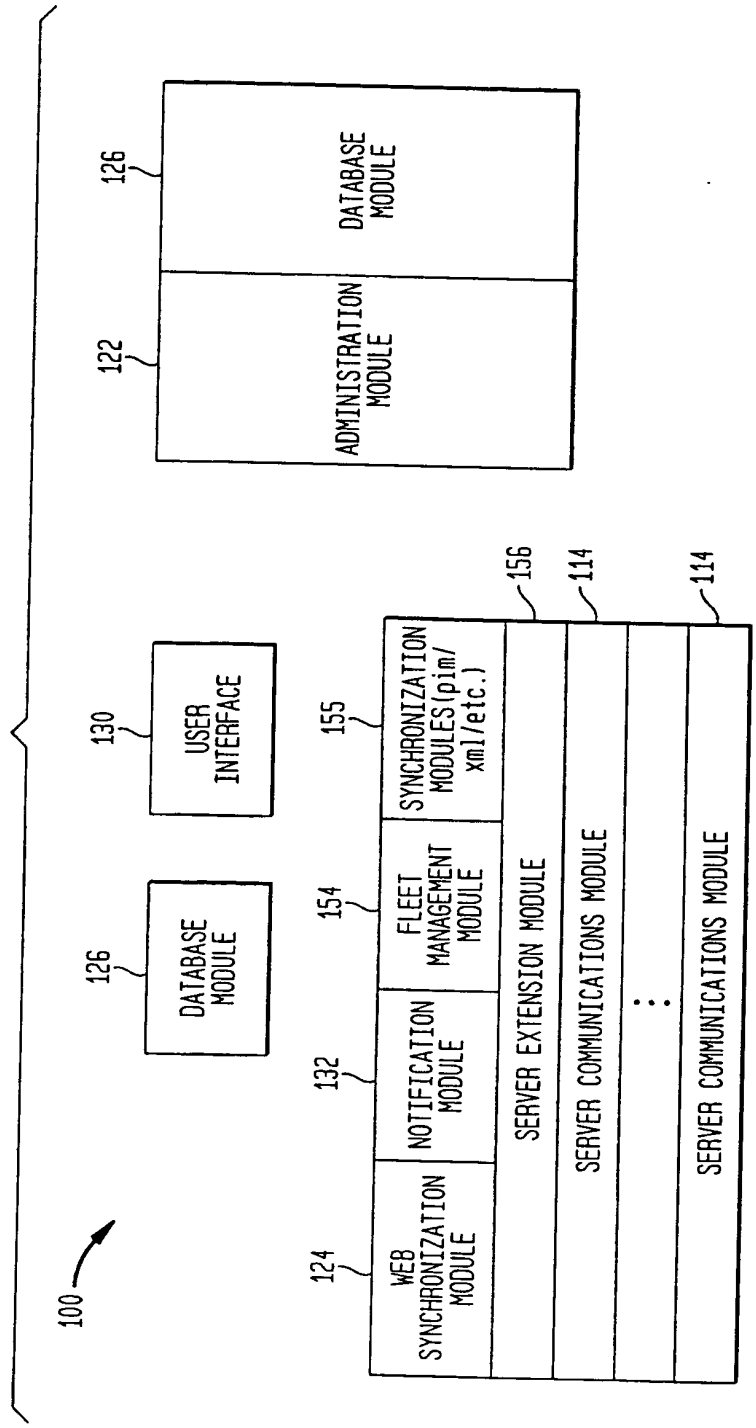

100
WEB SYNCHRONIZATION MODULE
124
NOTIFICATION MODULE
132
FLEET MANAGEMENT MODULE
154
SYNCHRONIZATION MODULES(pim/ xml/etc.)
155
SERVER EXTENSION MODULE
SERVER COMMUNICATIONS MODULE
SERVER COMMUNICATIONS MODULE
156
114
114
DATABASE MODULE
126
USER INTERFACE
130
ADMINISTRATION MODULE
122
DATABASE MODULE
126

FIG. 1B1

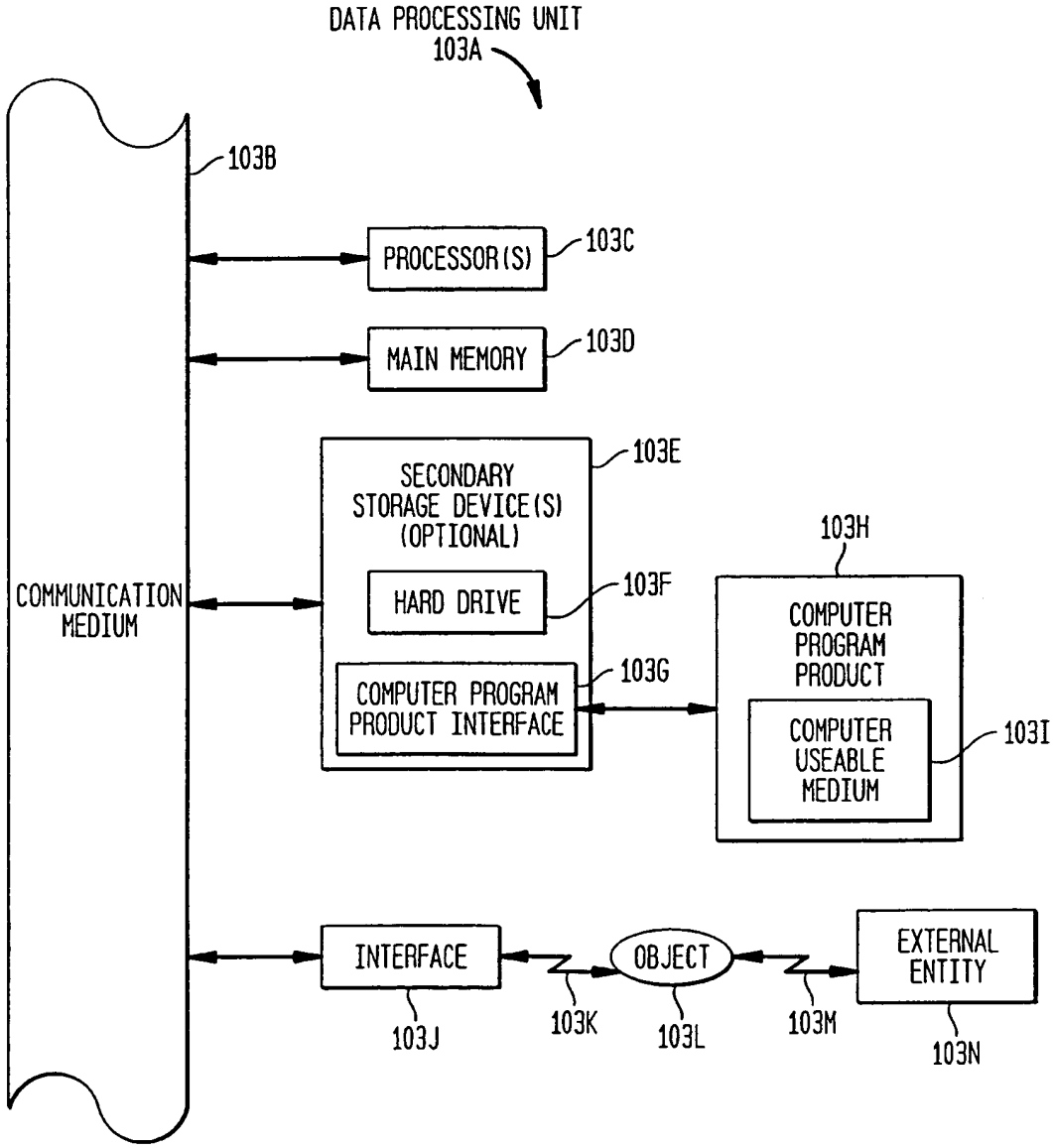

DATA PROCESSING UNIT
103A
103B
PROCESSOR(S)
103C
MAIN MEMORY
103D
SECONDARY STORAGE DEVICE(S) (OPTIONAL)
103E
HARD DRIVE
103F
COMPUTER PROGRAM PRODUCT INTERFACE
103G
COMMUNICATION MEDIUM
103H
COMPUTER PROGRAM PRODUCT
COMPUTER USEABLE MEDIUM
103I
INTERFACE
103J
103K
OBJECT
103L
103M
EXTERNAL ENTITY
103N

FIG. 1C

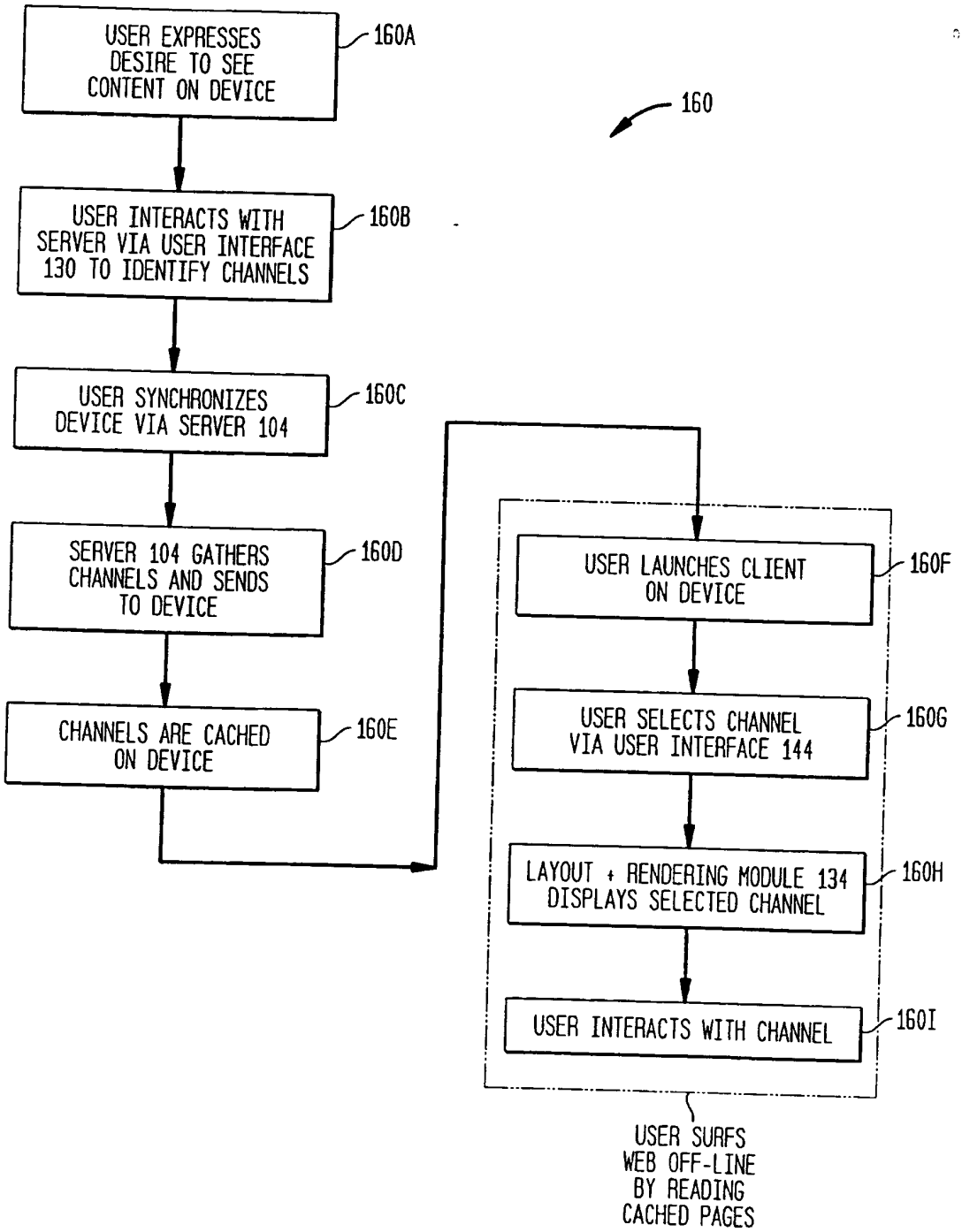

USER EXPRESSES DESIRE TO SEE CONTENT ON DEVICE
160A
160
USER INTERACTS WITH SERVER VIA USER INTERFACE 130 TO IDENTIFY CHANNELS
160B
USER SYNCHRONIZES DEVICE VIA SERVER 104
160C
SERVER 104 GATHERS CHANNELS AND SENDS TO DEVICE
160D
CHANNELS ARE CACHED ON DEVICE
160E
USER LAUNCHES CLIENT ON DEVICE
160F
USER SELECTS CHANNEL VIA USER INTERFACE 144
160G
LAYOUT + RENDERING MODULE 134 DISPLAYS SELECTED CHANNEL
160H
USER INTERACTS WITH CHANNEL
160I
USER SURFS WEB OFF-LINE BY READING CACHED PAGES

FIG. 1D

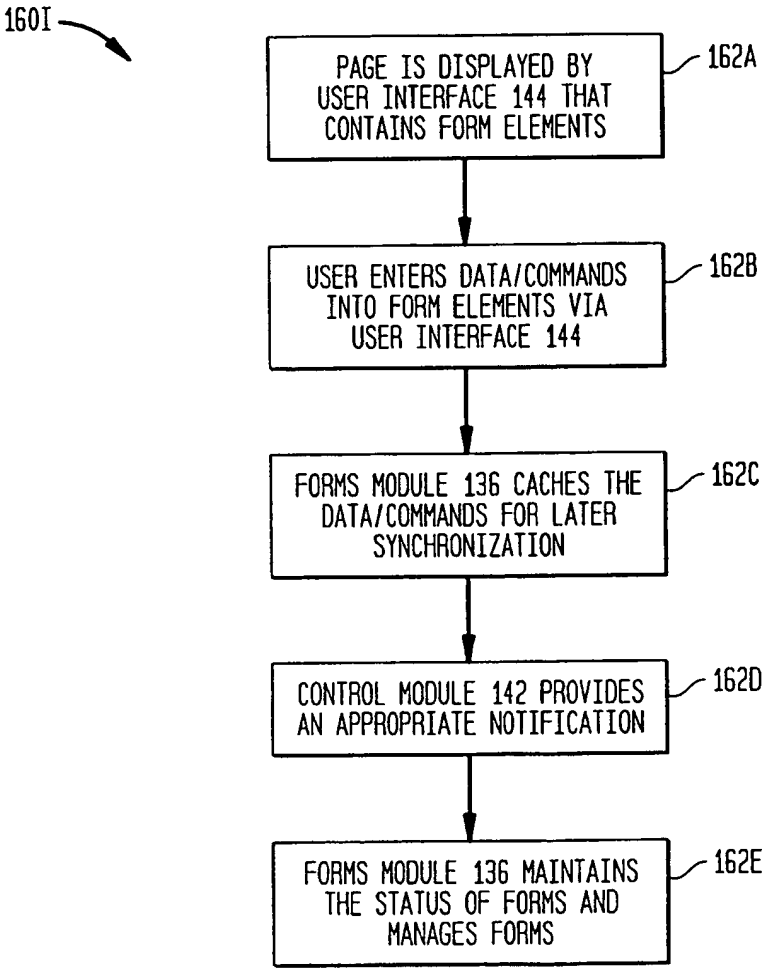

160I
PAGE IS DISPLAYED BY USER INTERFACE 144 THAT CONTAINS FORM ELEMENTS
162A
USER ENTERS DATA/COMMANDS INTO FORM ELEMENTS VIA USER INTERFACE 144
162B
FORMS MODULE 136 CACHES THE DATA/COMMANDS FOR LATER SYNCHRONIZATION
162C
CONTROL MODULE 142 PROVIDES AN APPROPRIATE NOTIFICATION
162D
FORMS MODULE 136 MAINTAINS THE STATUS OF FORMS AND MANAGES FORMS
162E

FIG. 1E

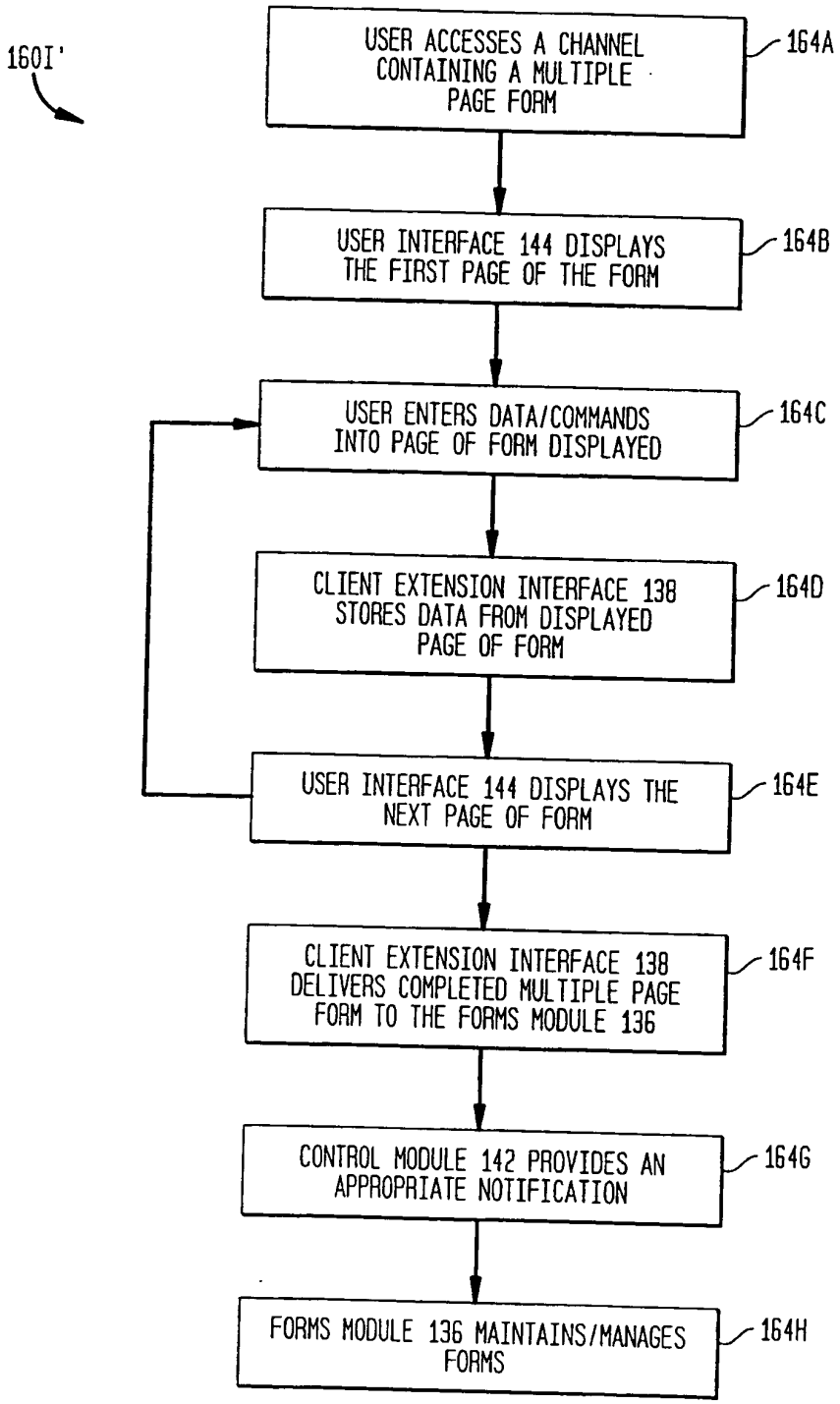

160I'
USER ACCESSES A CHANNEL CONTAINING A MULTIPLE PAGE FORM
164A
USER INTERFACE 144 DISPLAYS THE FIRST PAGE OF THE FORM
164B
USER ENTERS DATA/COMMANDS INTO PAGE OF FORM DISPLAYED
164C
CLIENT EXTENSION INTERFACE 138 STORES DATA FROM DISPLAYED PAGE OF FORM
164D
USER INTERFACE 144 DISPLAYS THE NEXT PAGE OF FORM
164E
CLIENT EXTENSION INTERFACE 138 DELIVERS COMPLETED MULTIPLE PAGE FORM TO THE FORMS MODULE 136
164F
CONTROL MODULE 142 PROVIDES AN APPROPRIATE NOTIFICATION
164G
FORMS MODULE 136 MAINTAINS/MANAGES FORMS
164H

FIG. 1F1

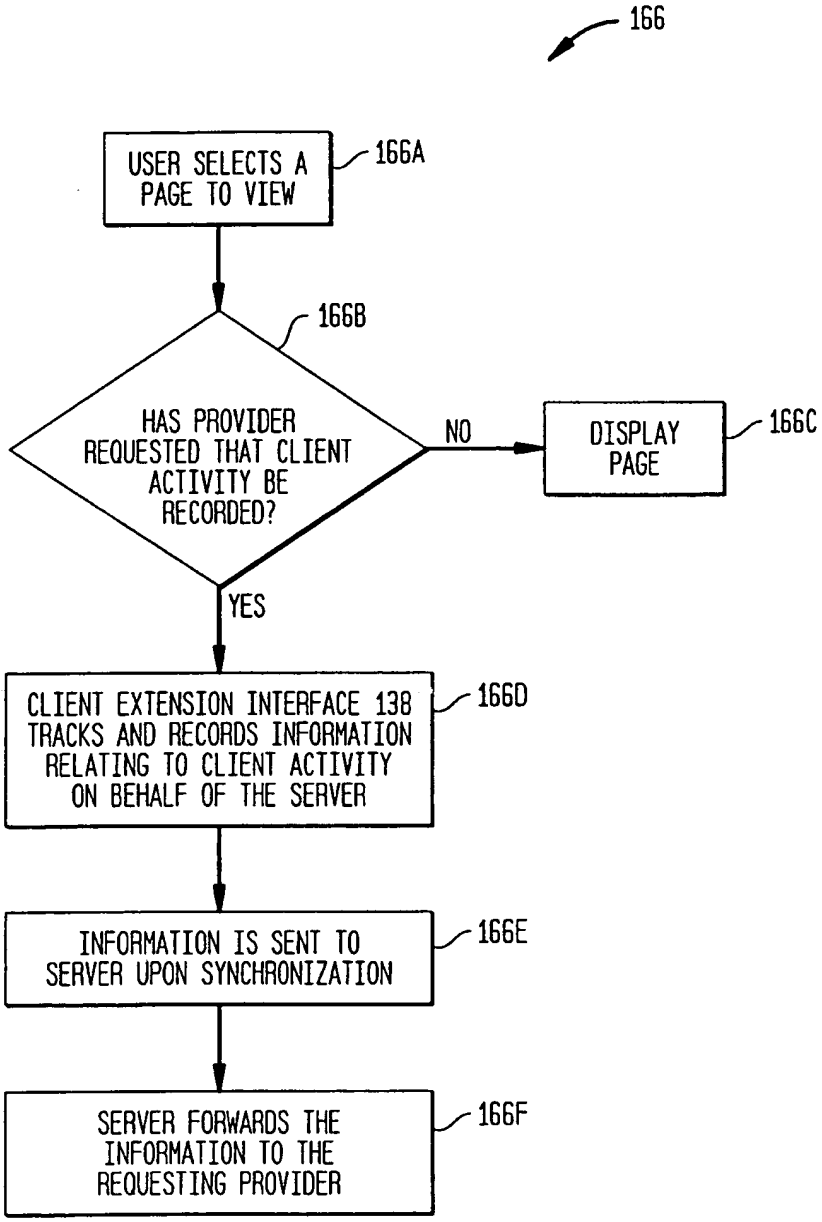

166
USER SELECTS A PAGE TO VIEW
166A
166B
HAS PROVIDER REQUESTED THAT CLIENT ACTIVITY BE RECORDED?
NO
DISPLAY PAGE
166C
YES
CLIENT EXTENSION INTERFACE 138 TRACKS AND RECORDS INFORMATION RELATING TO CLIENT ACTIVITY ON BEHALF OF THE SERVER
166D
INFORMATION IS SENT TO SERVER UPON SYNCHRONIZATION
166E
SERVER FORWARDS THE INFORMATION TO THE REQUESTING PROVIDER
166F

FIG. 1F2

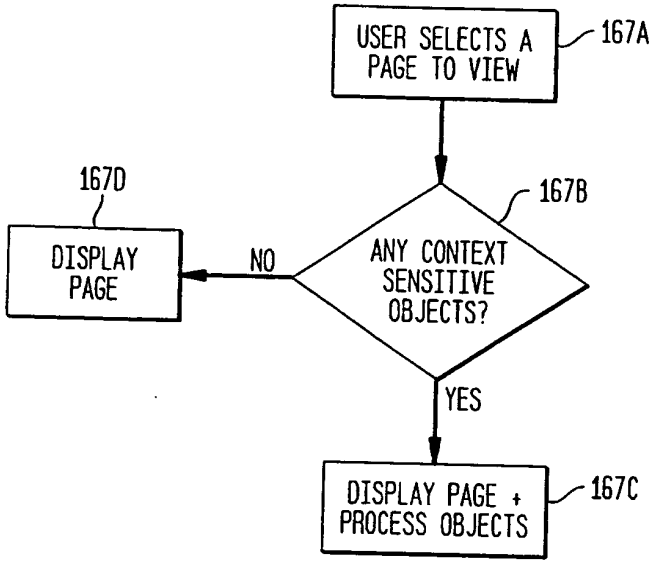


FIG. 1G

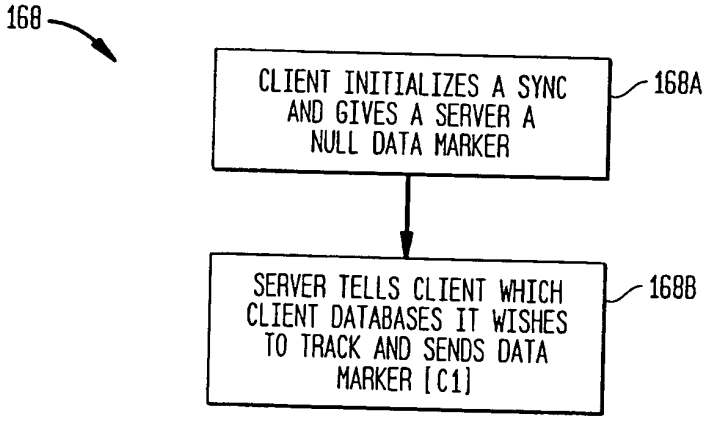


FIG. 1H1

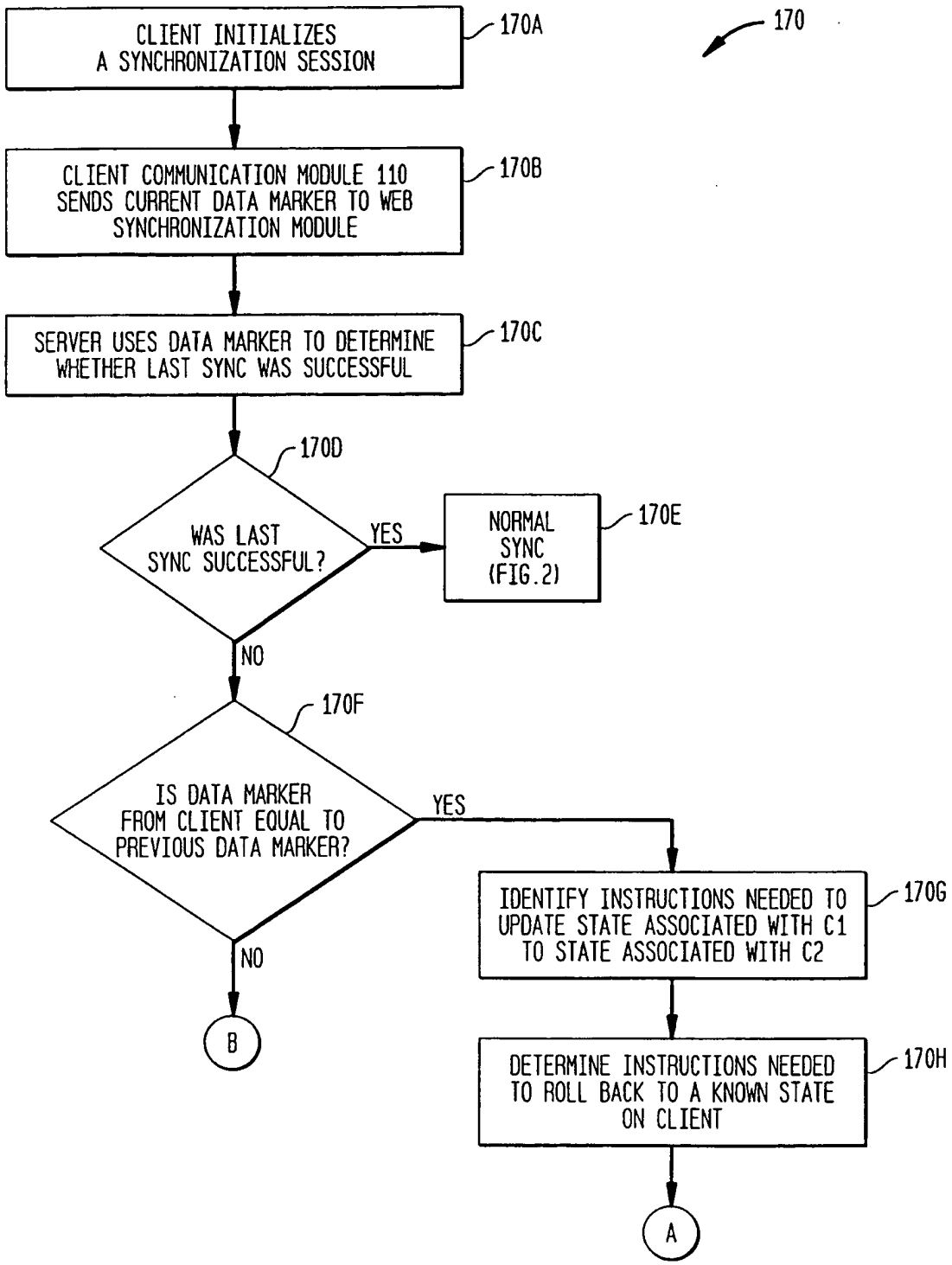

170
CLIENT INITIALIZES A SYNCHRONIZATION SESSION
170A
CLIENT COMMUNICATION MODULE 110 SENDS CURRENT DATA MARKER TO WEB SYNCHRONIZATION MODULE
170B
SERVER USES DATA MARKER TO DETERMINE WHETHER LAST SYNC WAS SUCCESSFUL
170C
170D
WAS LAST SYNC SUCCESSFUL?
YES
NORMAL SYNC (FIG.2)
170E
NO
170F
IS DATA MARKER FROM CLIENT EQUAL TO PREVIOUS DATA MARKER?
YES
NO
B
IDENTIFY INSTRUCTIONS NEEDED TO UPDATE STATE ASSOCIATED WITH C1 TO STATE ASSOCIATED WITH C2
170G
DETERMINE INSTRUCTIONS NEEDED TO ROLL BACK TO A KNOWN STATE ON CLIENT
170H
A

FIG. 1H2

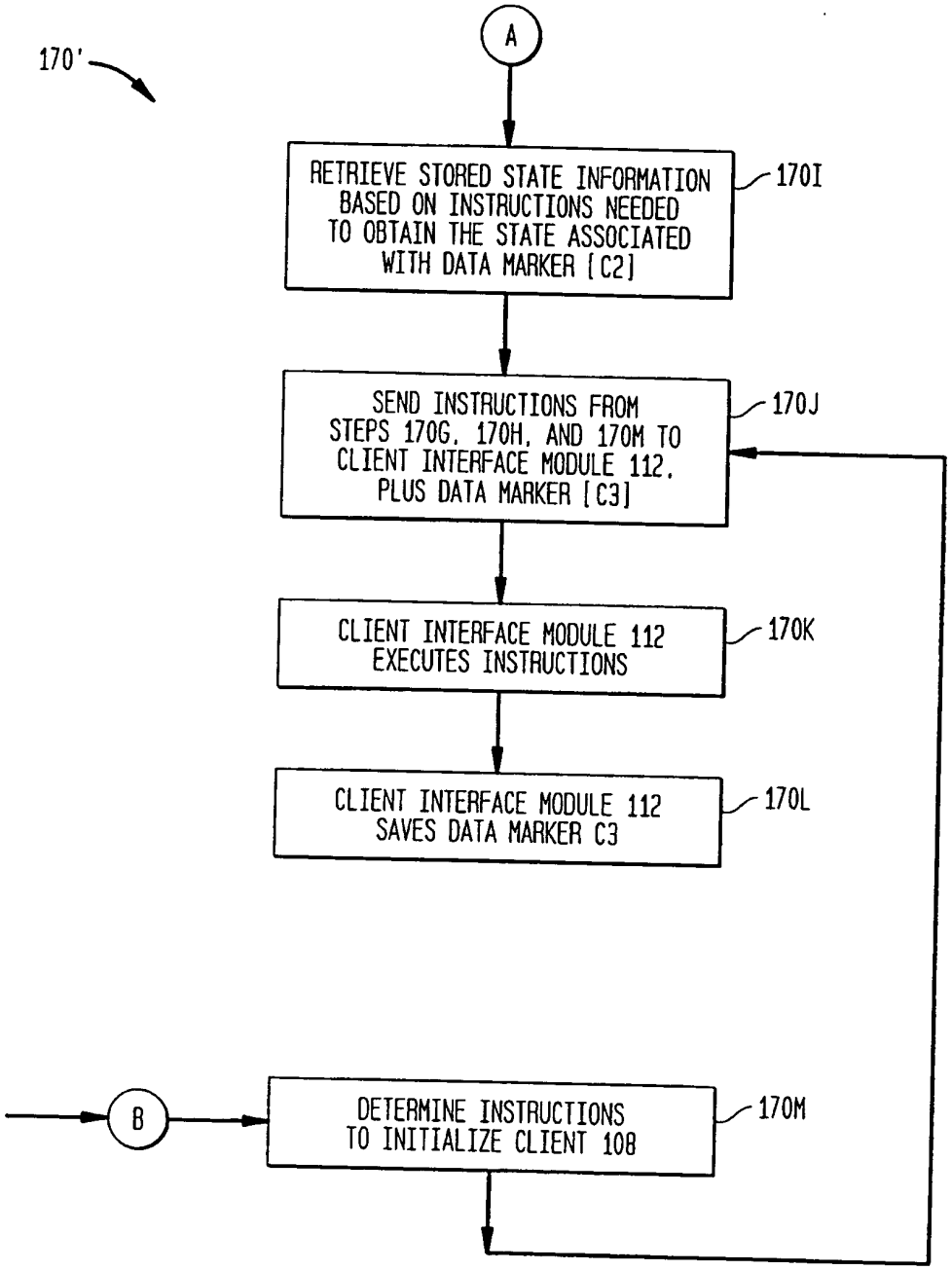

170'
A
RETRIEVE STORED STATE INFORMATION BASED ON INSTRUCTIONS NEEDED TO OBTAIN THE STATE ASSOCIATED WITH DATA MARKER [C2]
170I
SEND INSTRUCTIONS FROM STEPS 170G, 170H, AND 170M TO CLIENT INTERFACE MODULE 112, PLUS DATA MARKER [C3]
170J
CLIENT INTERFACE MODULE 112 EXECUTES INSTRUCTIONS
170K
CLIENT INTERFACE MODULE 112 SAVES DATA MARKER C3
170L
B
DETERMINE INSTRUCTIONS TO INITIALIZE CLIENT 108
170M

FIG. 1I1

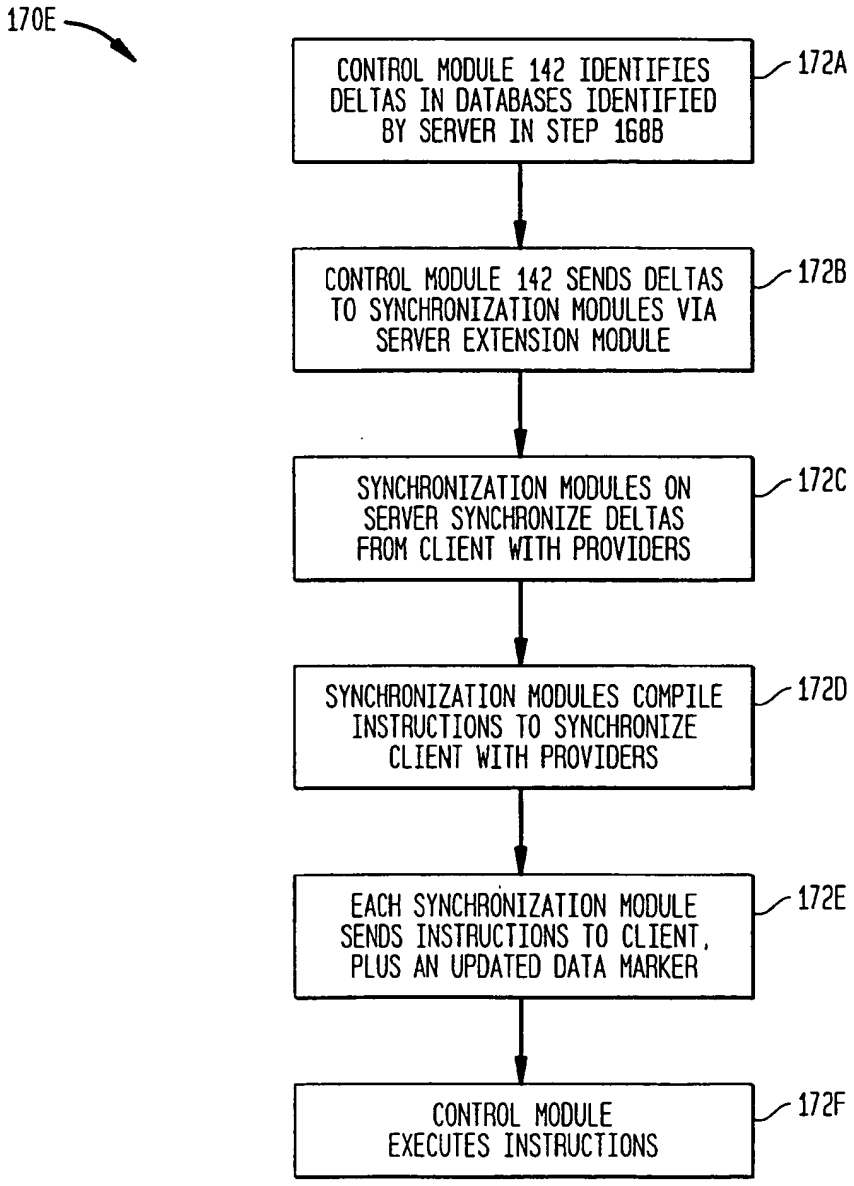

170E
CONTROL MODULE 142 IDENTIFIES DELTAS IN DATABASES IDENTIFIED BY SERVER IN STEP 168B
172A
CONTROL MODULE 142 SENDS DELTAS TO SYNCHRONIZATION MODULES VIA SERVER EXTENSION MODULE
172B
SYNCHRONIZATION MODULES ON SERVER SYNCHRONIZE DELTAS FROM CLIENT WITH PROVIDERS
172C
SYNCHRONIZATION MODULES COMPILE INSTRUCTIONS TO SYNCHRONIZE CLIENT WITH PROVIDERS
172D
EACH SYNCHRONIZATION MODULE SENDS INSTRUCTIONS TO CLIENT, PLUS AN UPDATED DATA MARKER
172E
CONTROL MODULE EXECUTES INSTRUCTIONS
172F

FIG. 1I2

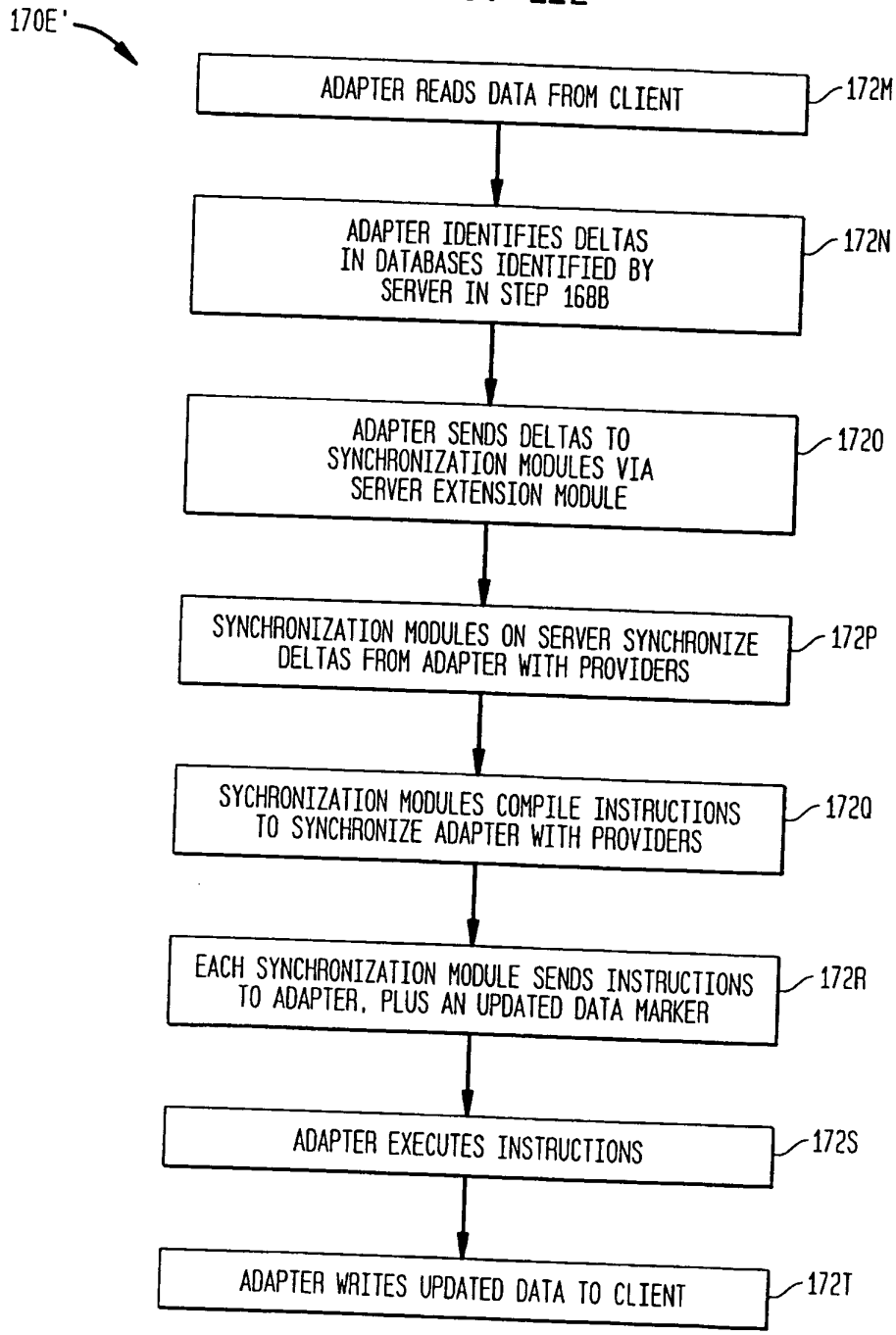

170E'
ADAPTER READS DATA FROM CLIENT
172M
ADAPTER IDENTIFIES DELTAS IN DATABASES IDENTIFIED BY SERVER IN STEP 168B
172N
ADAPTER SENDS DELTAS TO SYNCHRONIZATION MODULES VIA SERVER EXTENSION MODULE
172O
SYNCHRONIZATION MODULES ON SERVER SYNCHRONIZE DELTAS FROM ADAPTER WITH PROVIDERS
172P
SYCHRONIZATION MODULES COMPILE INSTRUCTIONS TO SYNCHRONIZE ADAPTER WITH PROVIDERS
172Q
EACH SYNCHRONIZATION MODULE SENDS INSTRUCTIONS TO ADAPTER, PLUS AN UPDATED DATA MARKER
172R
ADAPTER EXECUTES INSTRUCTIONS
172S
ADAPTER WRITES UPDATED DATA TO CLIENT
172T

FIG. 1J

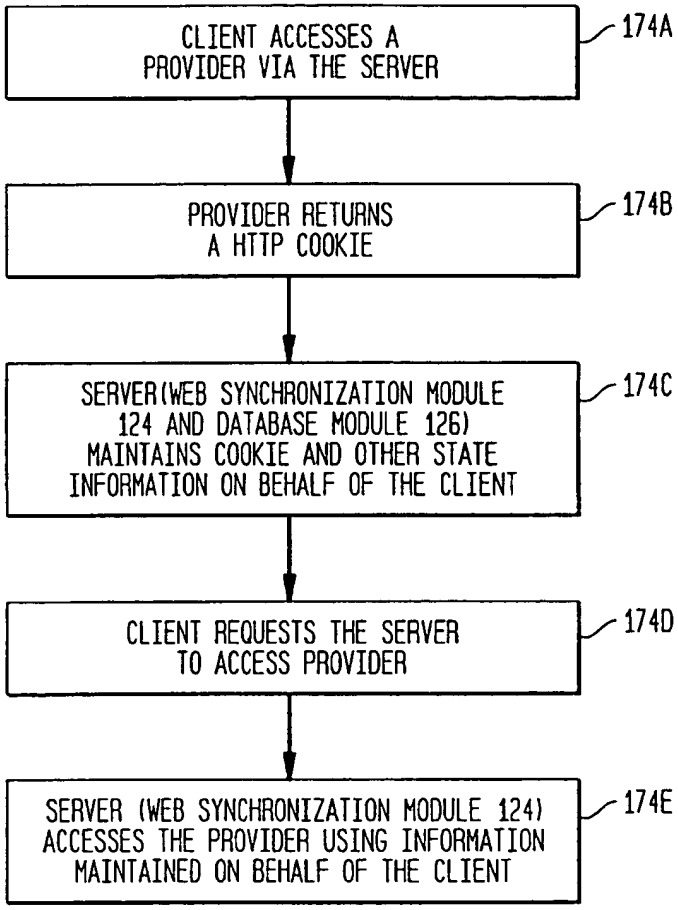

CLIENT ACCESSES A PROVIDER VIA THE SERVER
174A
PROVIDER RETURNS A HTTP COOKIE
174B
SERVER(WEB SYNCHRONIZATION MODULE 124 AND DATABASE MODULE 126) MAINTAINS COOKIE AND OTHER STATE INFORMATION ON BEHALF OF THE CLIENT
174C
CLIENT REQUESTS THE SERVER TO ACCESS PROVIDER
174D
SERVER (WEB SYNCHRONIZATION MODULE 124) ACCESSES THE PROVIDER USING INFORMATION MAINTAINED ON BEHALF OF THE CLIENT
174E

FIG. 1K

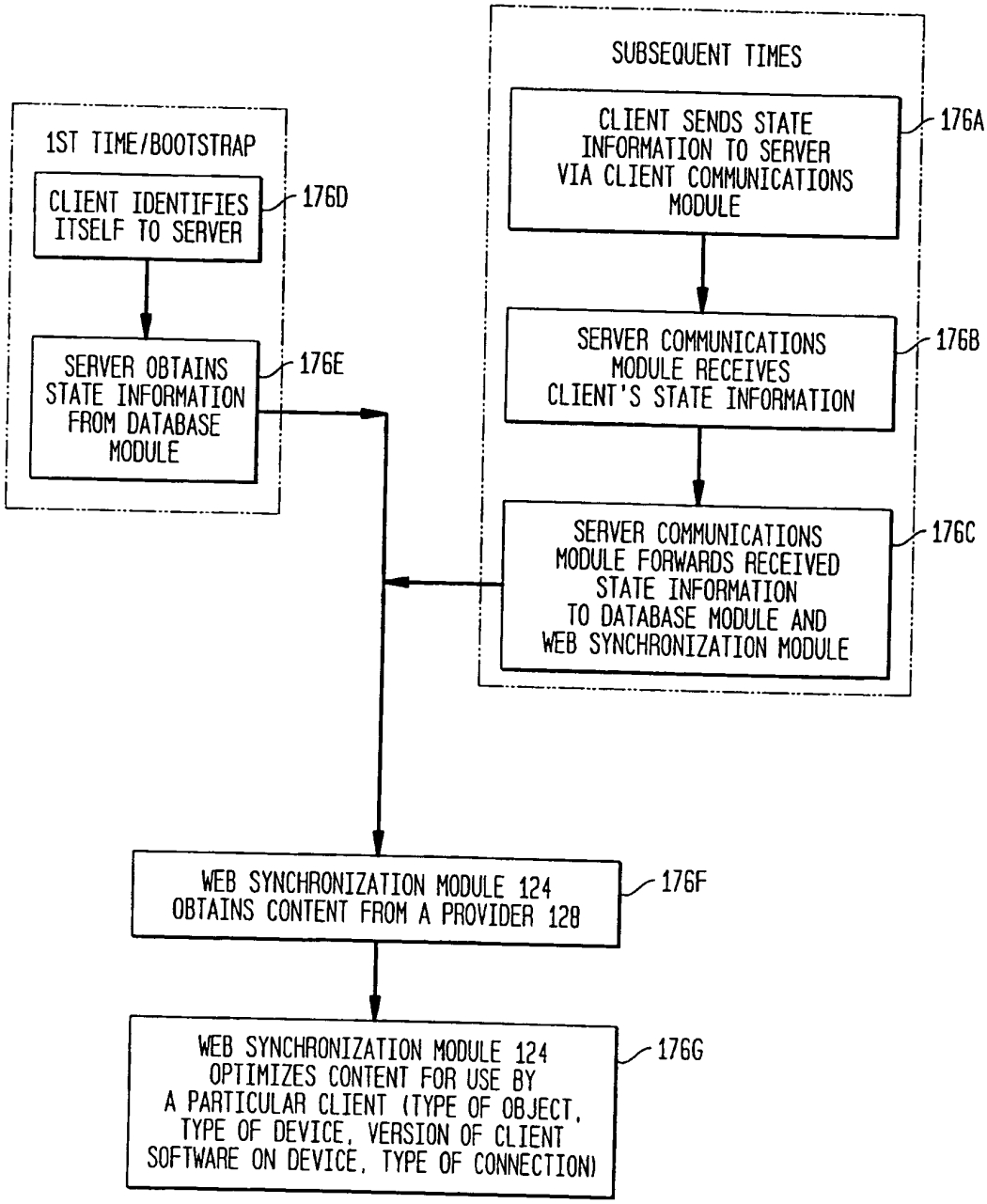

SUBSEQUENT TIMES
CLIENT SENDS STATE INFORMATION TO SERVER VIA CLIENT COMMUNICATIONS MODULE
176A
SERVER COMMUNICATIONS MODULE RECEIVES CLIENT'S STATE INFORMATION
176B
SERVER COMMUNICATIONS MODULE FORWARDS RECEIVED STATE INFORMATION TO DATABASE MODULE AND WEB SYNCHRONIZATION MODULE
176C
1ST TIME/BOOTSTRAP
CLIENT IDENTIFIES ITSELF TO SERVER
176D
SERVER OBTAINS STATE INFORMATION FROM DATABASE MODULE
176E
WEB SYNCHRONIZATION MODULE 124 OBTAINS CONTENT FROM A PROVIDER 128
176F
WEB SYNCHRONIZATION MODULE 124 OPTIMIZES CONTENT FOR USE BY A PARTICULAR CLIENT (TYPE OF OBJECT, TYPE OF DEVICE, VERSION OF CLIENT SOFTWARE ON DEVICE, TYPE OF CONNECTION)
176G

FIG. 1L

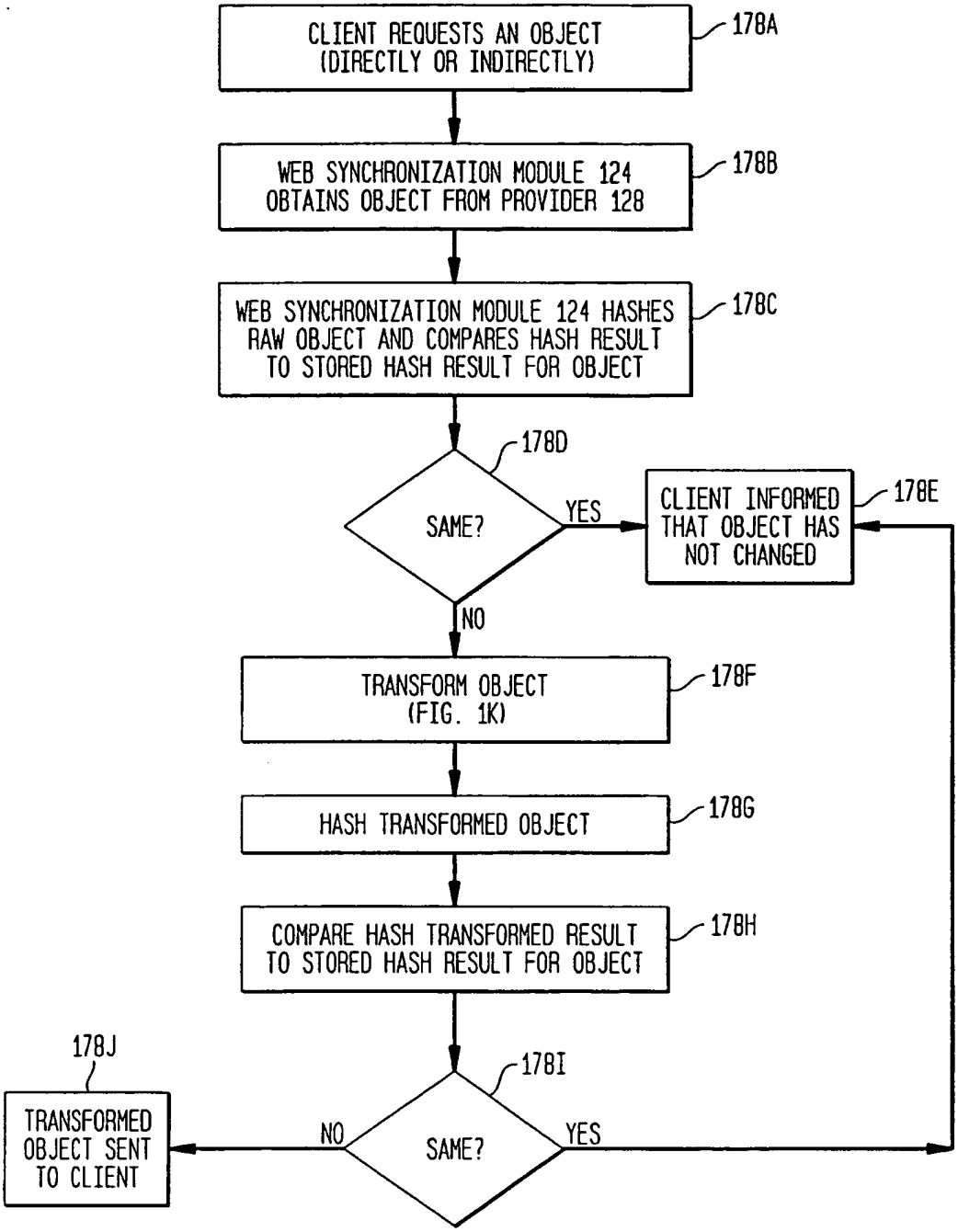
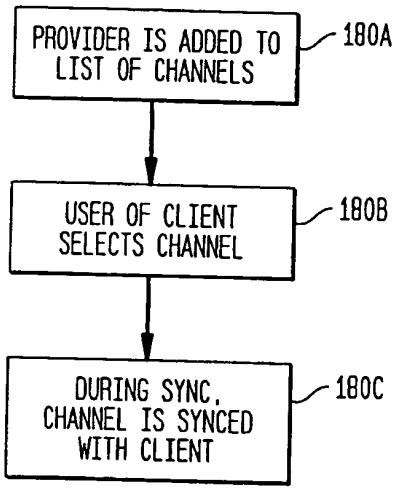

CLIENT REQUESTS AN OBJECT (DIRECTLY OR INDIRECTLY)
178A
WEB SYNCHRONIZATION MODULE 124 OBTAINS OBJECT FROM PROVIDER 128
178B
WEB SYNCHRONIZATION MODULE 124 HASHES RAW OBJECT AND COMPARES HASH RESULT TO STORED HASH RESULT FOR OBJECT
178C
178D
SAME?
YES
CLIENT INFORMED THAT OBJECT HAS NOT CHANGED
178E
NO
TRANSFORM OBJECT (FIG. 1K)
178F
HASH TRANSFORMED OBJECT
178G
COMPARE HASH TRANSFORMED RESULT TO STORED HASH RESULT FOR OBJECT
178H
178J
178I
TRANSFORMED OBJECT SENT TO CLIENT
NO
SAME?
YES

FIG. 1M



- MUSIC
- PHOTO ALBUMS
- BOOKS
- MOVIES
- ETC.

FIG. 1N

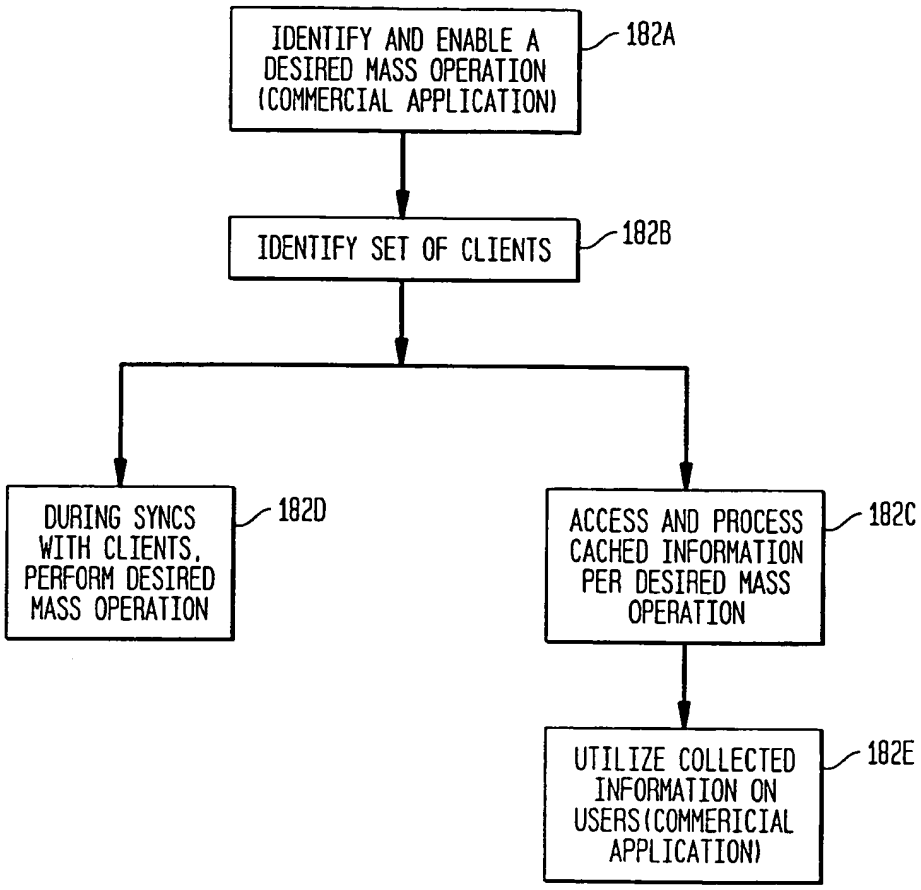

IDENTIFY AND ENABLE A DESIRED MASS OPERATION (COMMERCIAL APPLICATION)
182A
IDENTIFY SET OF CLIENTS
182B
DURING SYNCS WITH CLIENTS, PERFORM DESIRED MASS OPERATION
182D
ACCESS AND PROCESS CACHED INFORMATION PER DESIRED MASS OPERATION
182C
UTILIZE COLLECTED INFORMATION ON USERS(COMMERICIAL APPLICATION)
182E

FIG. 10

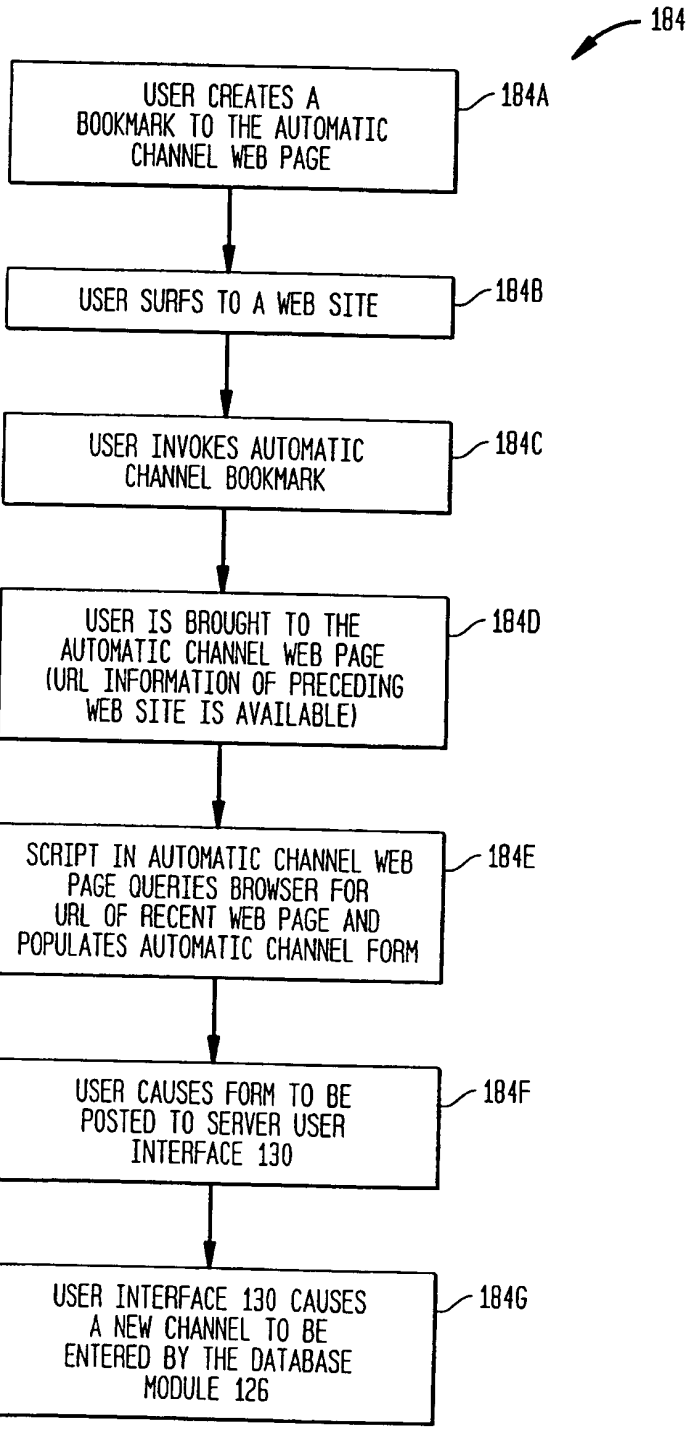

184
USER CREATES A BOOKMARK TO THE AUTOMATIC CHANNEL WEB PAGE
184A
USER SURFS TO A WEB SITE
184B
USER INVOKES AUTOMATIC CHANNEL BOOKMARK
184C
USER IS BROUGHT TO THE AUTOMATIC CHANNEL WEB PAGE (URL INFORMATION OF PRECEDING WEB SITE IS AVAILABLE)
184D
SCRIPT IN AUTOMATIC CHANNEL WEB PAGE QUERIES BROWSER FOR URL OF RECENT WEB PAGE AND POPULATES AUTOMATIC CHANNEL FORM
184E
USER CAUSES FORM TO BE POSTED TO SERVER USER INTERFACE 130
184F
USER INTERFACE 130 CAUSES A NEW CHANNEL TO BE ENTERED BY THE DATABASE MODULE 126
184G

FIG. 1P

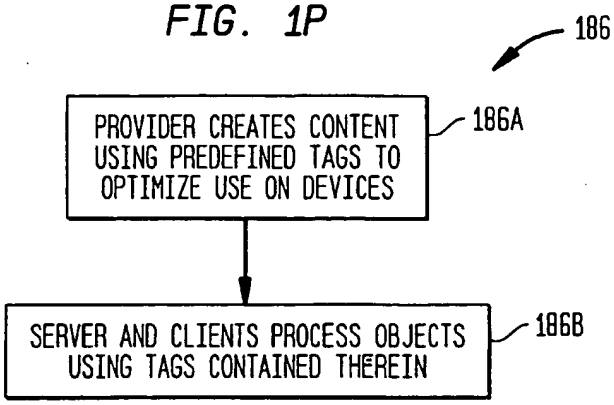


FIG. 1Q

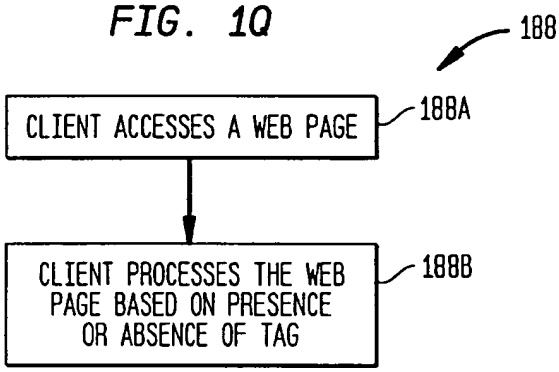


FIG. 1R

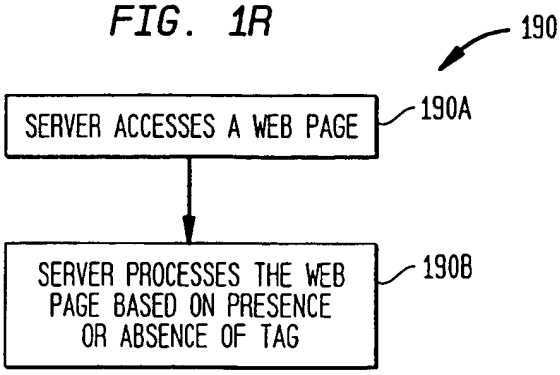


FIG. 1S

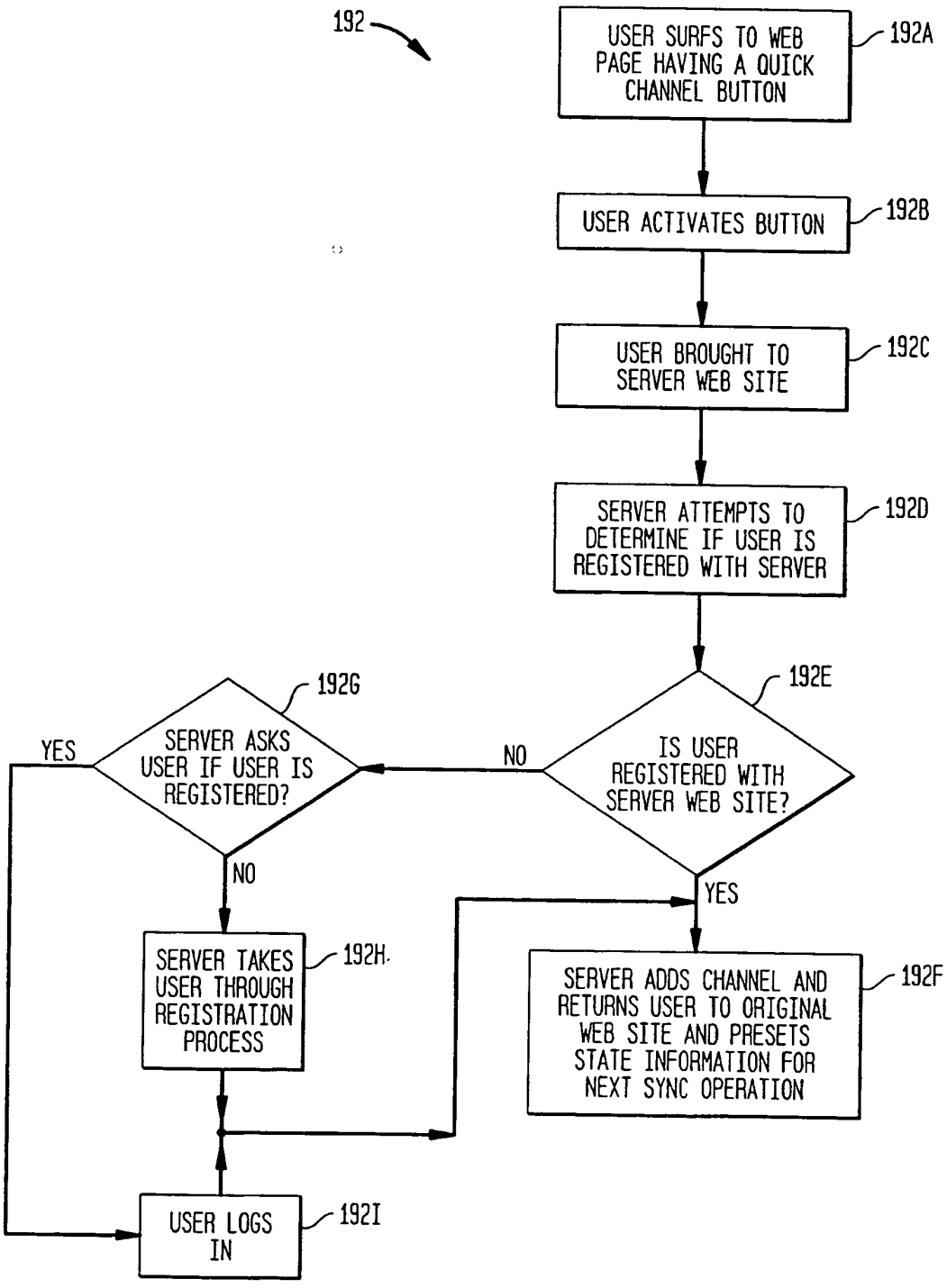

192
USER SURFS TO WEB PAGE HAVING A QUICK CHANNEL BUTTON
192A
USER ACTIVATES BUTTON
192B
USER BROUGHT TO SERVER WEB SITE
192C
SERVER ATTEMPTS TO DETERMINE IF USER IS REGISTERED WITH SERVER
192D
IS USER REGISTERED WITH SERVER WEB SITE?
192E
NO
YES
SERVER ASKS USER IF USER IS REGISTERED?
192G
YES
NO
SERVER ADDS CHANNEL AND RETURNS USER TO ORIGINAL WEB SITE AND PRESETS STATE INFORMATION FOR NEXT SYNC OPERATION
192F
SERVER TAKES USER THROUGH REGISTRATION PROCESS
192H
USER LOGS IN
192I

FIG. 1T

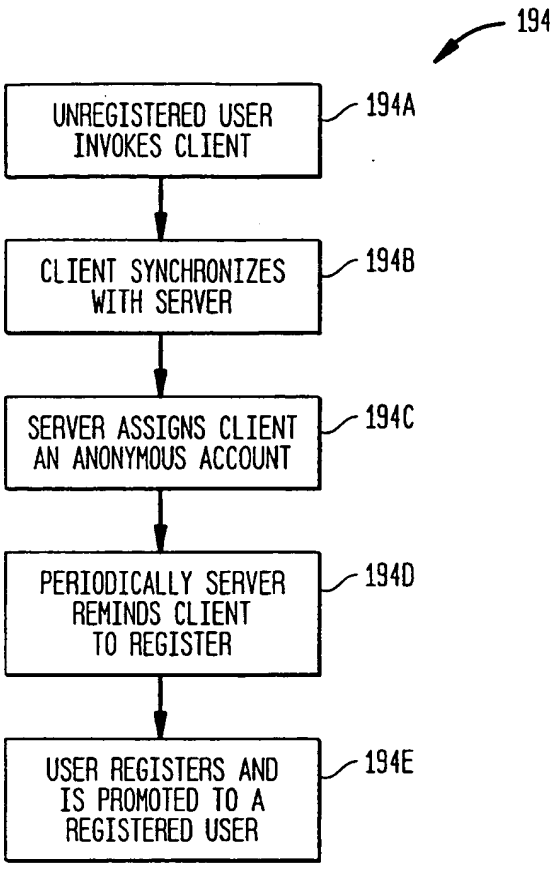

194
UNREGISTERED USER INVOKES CLIENT
194A
CLIENT SYNCHRONIZES WITH SERVER
194B
SERVER ASSIGNS CLIENT AN ANONYMOUS ACCOUNT
194C
PERIODICALLY SERVER REMINDS CLIENT TO REGISTER
194D
USER REGISTERS AND IS PROMOTED TO A REGISTERED USER
194E

FIG. 1U

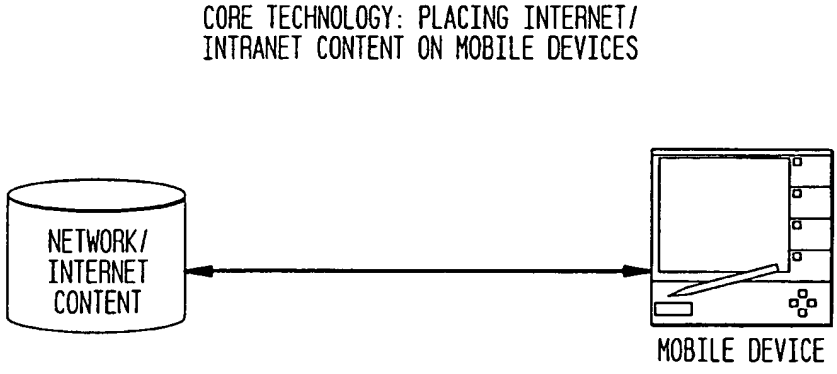

CORE TECHNOLOGY: PLACING INTERNET/ INTRANET CONTENT ON MOBILE DEVICES
NETWORK/ INTERNET CONTENT
MOBILE DEVICE

FIG. 1V

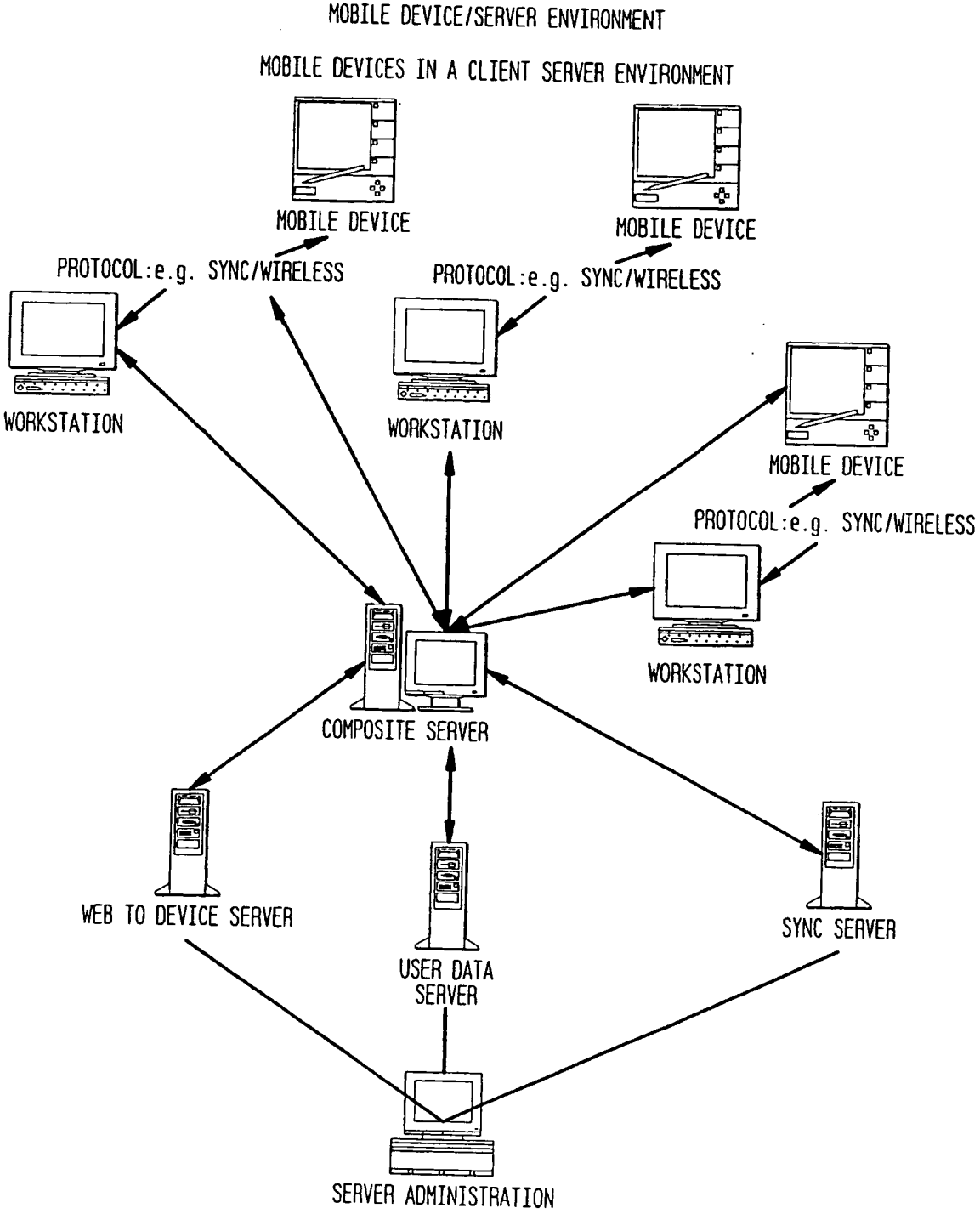

MOBILE DEVICE/SERVER ENVIRONMENT
MOBILE DEVICES IN A CLIENT SERVER ENVIRONMENT
MOBILE DEVICE
MOBILE DEVICE
PROTOCOL:e.g. SYNC/WIRELESS
PROTOCOL:e.g. SYNC/WIRELESS
WORKSTATION
WORKSTATION
MOBILE DEVICE
PROTOCOL:e.g. SYNC/WIRELESS
WORKSTATION
COMPOSITE SERVER
WEB TO DEVICE SERVER
USER DATA SERVER
SYNC SERVER
SERVER ADMINISTRATION

FIG. 1W

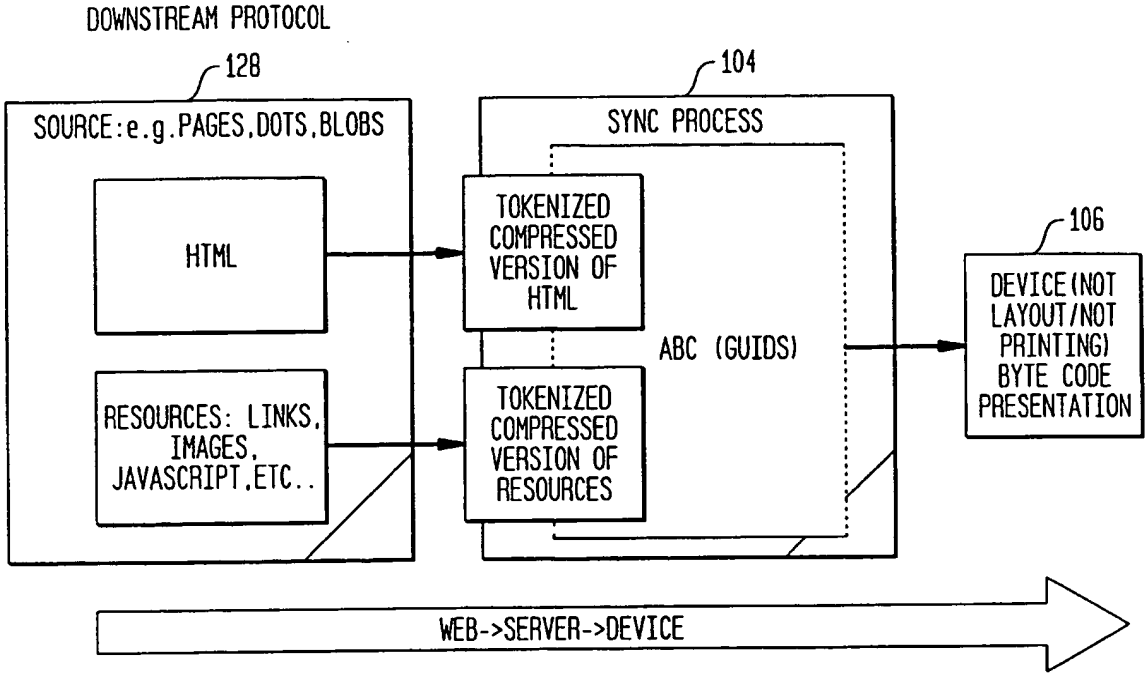

DOWNSTREAM PROTOCOL
128
SOURCE:e.g.PAGES,DOTS,BLOBS
HTML
RESOURCES: LINKS, IMAGES, JAVASCRIPT,ETC..
104
SYNC PROCESS
TOKENIZED COMPRESSED VERSION OF HTML
ABC (GUIDS)
TOKENIZED COMPRESSED VERSION OF RESOURCES
106
DEVICE(NOT LAYOUT/NOT PRINTING) BYTE CODE PRESENTATION
WEB->SERVER->DEVICE

FIG. 1X

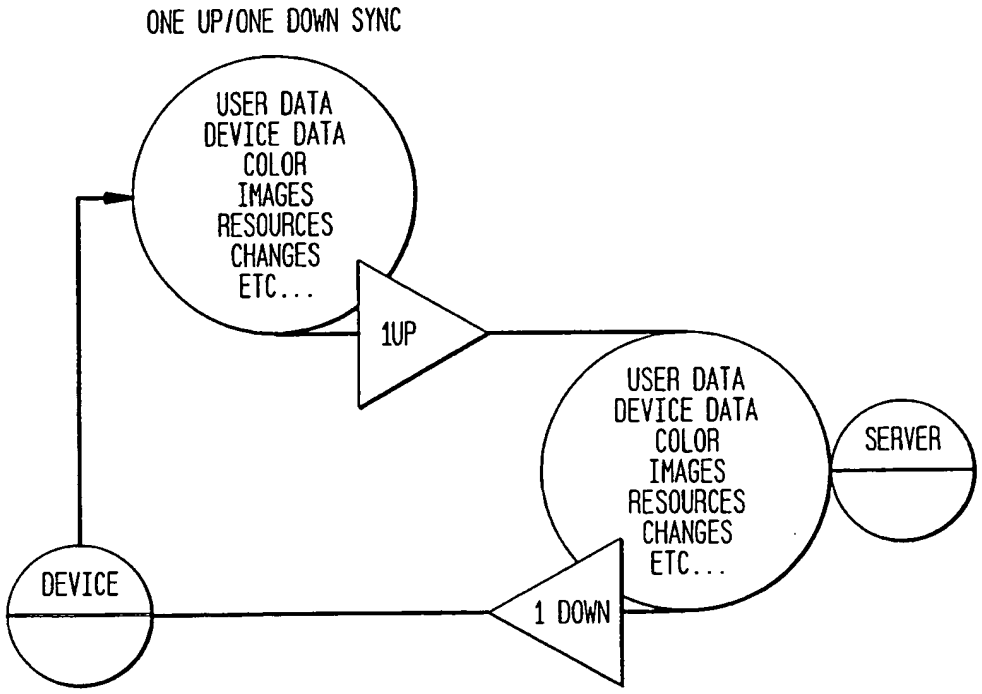
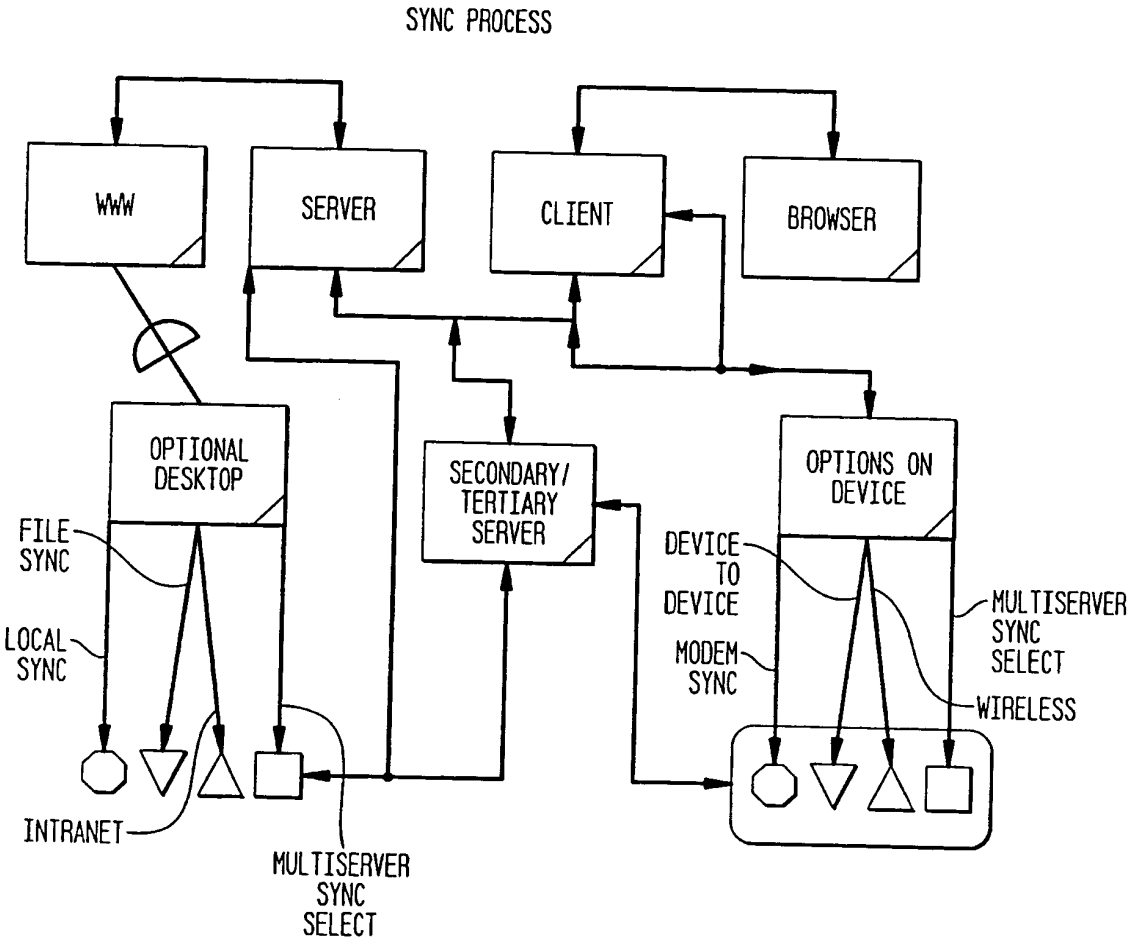

ONE UP/ONE DOWN SYNC
USER DATA
DEVICE DATA
COLOR
IMAGES
RESOURCES
CHANGES
ETC...
1UP
USER DATA
DEVICE DATA
COLOR
IMAGES
RESOURCES
CHANGES
ETC...
SERVER
DEVICE
1 DOWN

FIG. 1Y


SYNC PROCESS
WWW
SERVER
CLIENT
BROWSER
OPTIONAL DESKTOP
SECONDARY/ TERTIARY SERVER
OPTIONS ON DEVICE
FILE SYNC
LOCAL SYNC
INTRANET
MULTISERVER SYNC SELECT
DEVICE TO DEVICE
MODEM SYNC
MULTISERVER SYNC SELECT
WIRELESS

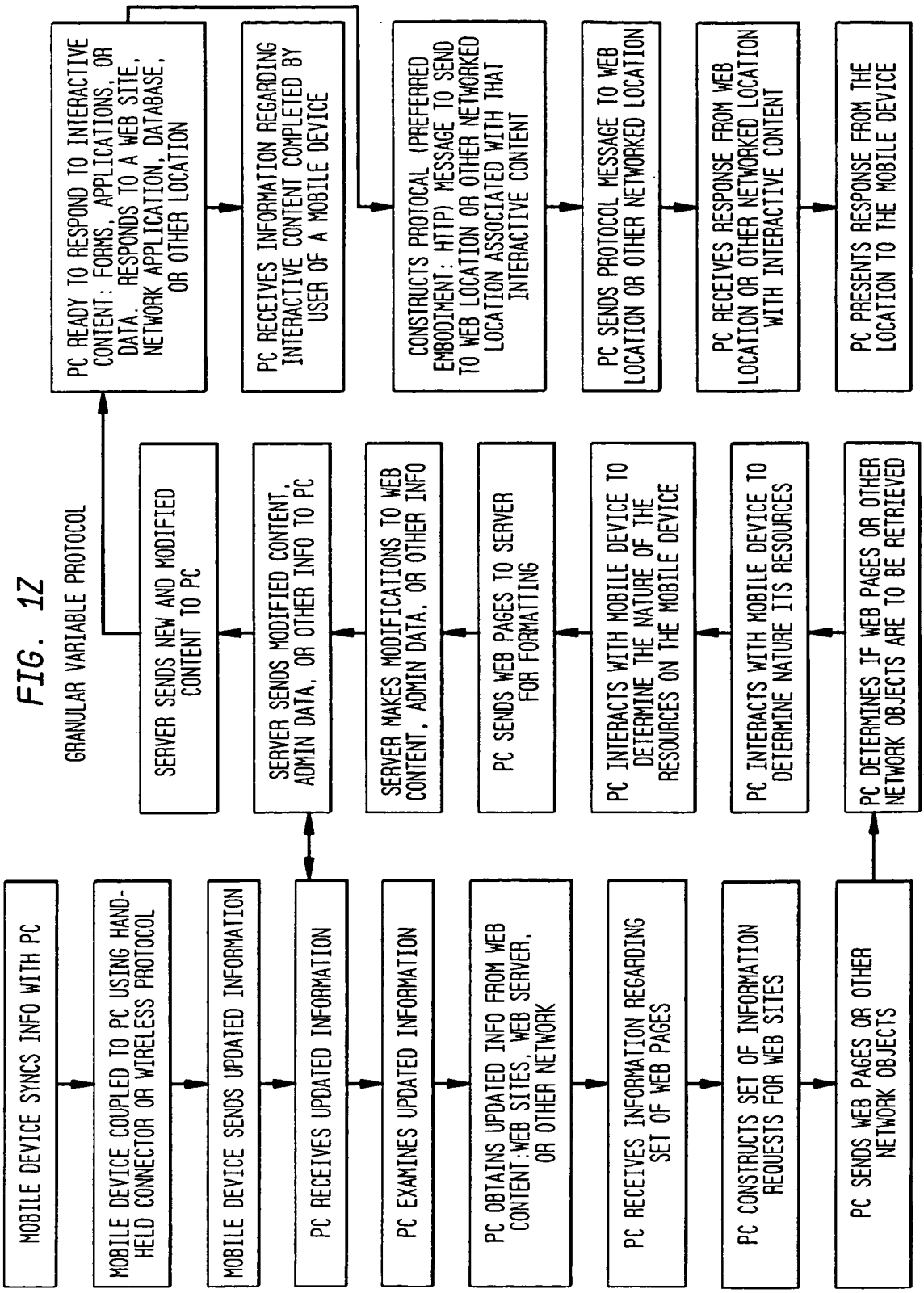

MOBILE DEVICE SYNCS INFO WITH PC
MOBILE DEVICE COUPLED TO PC USING HAND-HELD CONNECTOR OR WIRELESS PROTOCOL
MOBILE DEVICE SENDS UPDATED INFORMATION
PC RECEIVES UPDATED INFORMATION
PC EXAMINES UPDATED INFORMATION
PC OBTAINS UPDATED INFO FROM WEB CONTENT: WEB SITES, WEB SERVER, OR OTHER NETWORK
PC RECEIVES INFORMATION REGARDING SET OF WEB PAGES
PC CONSTRUCTS SET OF INFORMATION REQUESTS FOR WEB SITES
PC SENDS WEB PAGES OR OTHER NETWORK OBJECTS
PC DETERMINES IF WEB PAGES OR OTHER NETWORK OBJECTS ARE TO BE RETRIEVED
PC INTERACTS WITH MOBILE DEVICE TO DETERMINE NATURE ITS RESOURCES
PC INTERACTS WITH MOBILE DEVICE TO DETERMINE THE NATURE OF THE RESOURCES ON THE MOBILE DEVICE
PC SENDS WEB PAGES TO SERVER FOR FORMATTING
SERVER MAKES MODIFICATIONS TO WEB CONTENT, ADMIN DATA, OR OTHER INFO
SERVER SENDS MODIFIED CONTENT, ADMIN DATA, OR OTHER INFO TO PC
SERVER SENDS NEW AND MODIFIED CONTENT TO PC
PC PRESENTS RESPONSE FROM THE LOCATION TO THE MOBILE DEVICE
PC RECEIVES RESPONSE FROM WEB LOCATION OR OTHER NETWORKED LOCATION WITH INTERACTIVE CONTENT
PC SENDS PROTOCOL MESSAGE TO WEB LOCATION OR OTHER NETWORKED LOCATION
CONSTRUCTS PROTOCAL (PREFERRED EMBODIMENT: HTTP) MESSAGE TO SEND TO WEB LOCATION OR OTHER NETWORKED LOCATION ASSOCIATED WITH THAT INTERACTIVE CONTENT
PC RECEIVES INFORMATION REGARDING INTERACTIVE CONTENT COMPLETED BY USER OF A MOBILE DEVICE
PC READY TO RESPOND TO INTERACTIVE CONTENT: FORMS, APPLICATIONS, OR DATA. RESPONDS TO A WEB SITE, NETWORK APPLICATION, DATABASE, OR OTHER LOCATION
GRANULAR VARIABLE PROTOCOL

FIG. 1Z

FIG. 1AA

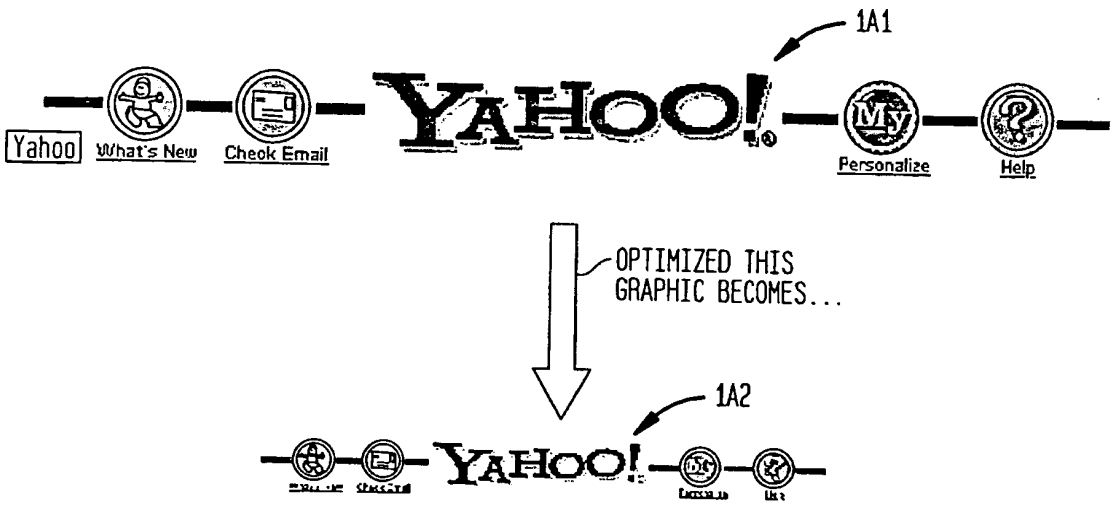

OPTMIZATION: AN EXAMPLE
1A1
Yahoo
What's New
Check Email
YAHOO!
My
Personalize
Help
OPTIMIZED THIS GRAPHIC BECOMES...
1A2
YAHOO!

FIG. 1AB

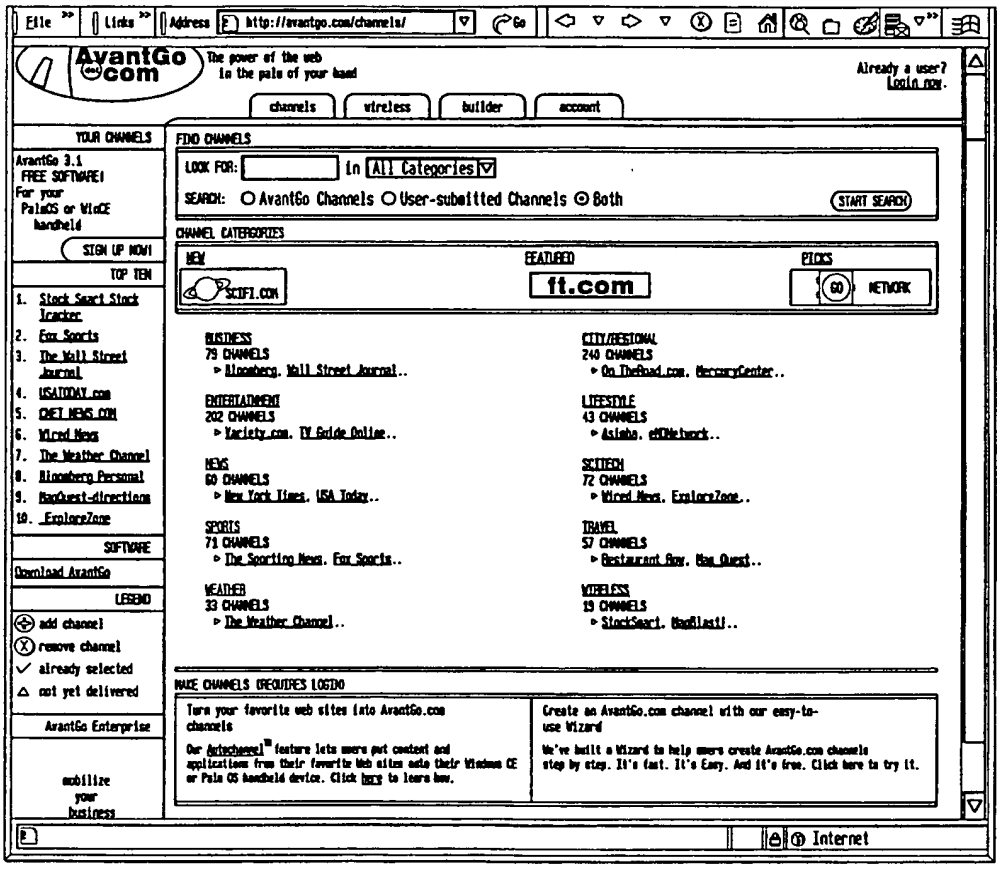

File
Links
Address
http://avantgo.com/channels/
Go
AvantGo .com
The power of the web in the palm of your hand
Already a user? Login now.
channels
wireless
builder
account
YOUR CHANNELS
AvantGo 3.1 FREE SOFTWARE! For your PalmOS or WinCE handheld
SIGN UP NOW!
TOP TEN
1. Stock Smart Stock Tracker
2. Fox Sports
3. The Wall Street Journal
4. USATODAY.com
5. CNET NEWS.COM
6. Wired News
7. The Weather Channel
8. Bloomberg Personal
9. MapQuest-directions
10. ExploreZone
SOFTWARE
Download AvantGo
LEGEND
add channel
remove channel
already selected
not yet delivered
AvantGo Enterprise
mobilize your business
FIND CHANNELS
LOOK FOR:
in All Categories
SEARCH: AvantGo Channels User-submitted Channels Both
START SEARCH
CHANNEL CATEGORIES
NEW
SCIFI.COM
FEATURED
ft.com
PICKS
GO NETWORK
BUSINESS
79 CHANNELS
Bloomberg, Wall Street Journal..
ENTERTAINMENT
202 CHANNELS
Variety.com, TV Guide Online..
NEWS
60 CHANNELS
New York Times, USA Today..
SPORTS
71 CHANNELS
The Sporting News, Fox Sports..
WEATHER
33 CHANNELS
The Weather Channel..
CITY/REGIONAL
240 CHANNELS
On The Road.com, MercuryCenter..
LIFESTYLE
43 CHANNELS
Asiaba, eWOnetwork..
SCITECH
72 CHANNELS
Wired News, ExploreZone..
TRAVEL
57 CHANNELS
Restaurant Row, Map Quest..
WIRELESS
19 CHANNELS
StockSmart, MapBlast!..
MAKE CHANNELS &REQUIRES LOGIN
Turn your favorite web sites into AvantGo.com channels
Our Avtochannel™ feature lets users get content and applications from their favorite Web sites onto their Windows CE or Palm OS handheld device. Click here to learn how.
Create an AvantGo.com channel with our easy-to-use Wizard
We've built a Wizard to help users create AvantGo.com channels step by step. It's fast. It's Easy. And it's free. Click here to try it.
Internet

FIG. 2

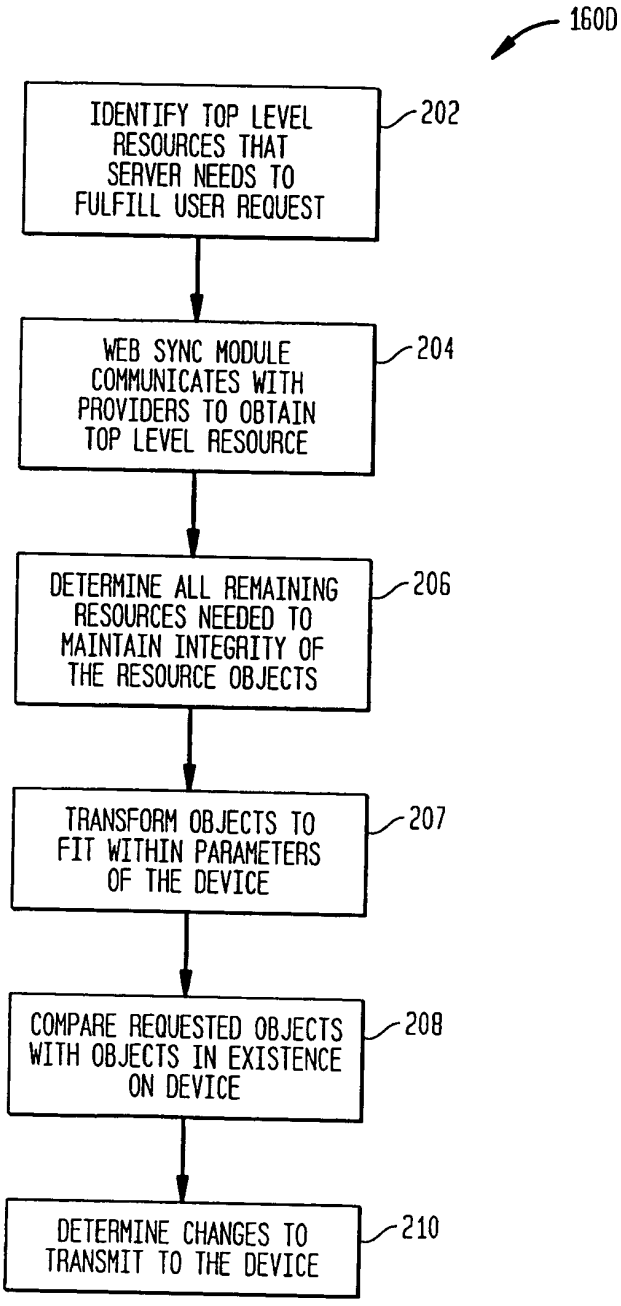

160D
IDENTIFY TOP LEVEL RESOURCES THAT SERVER NEEDS TO FULFILL USER REQUEST
202
WEB SYNC MODULE COMMUNICATES WITH PROVIDERS TO OBTAIN TOP LEVEL RESOURCE
204
DETERMINE ALL REMAINING RESOURCES NEEDED TO MAINTAIN INTEGRITY OF THE RESOURCE OBJECTS
206
TRANSFORM OBJECTS TO FIT WITHIN PARAMETERS OF THE DEVICE
207
COMPARE REQUESTED OBJECTS WITH OBJECTS IN EXISTENCE ON DEVICE
208
DETERMINE CHANGES TO TRANSMIT TO THE DEVICE
210

FIG. 3A

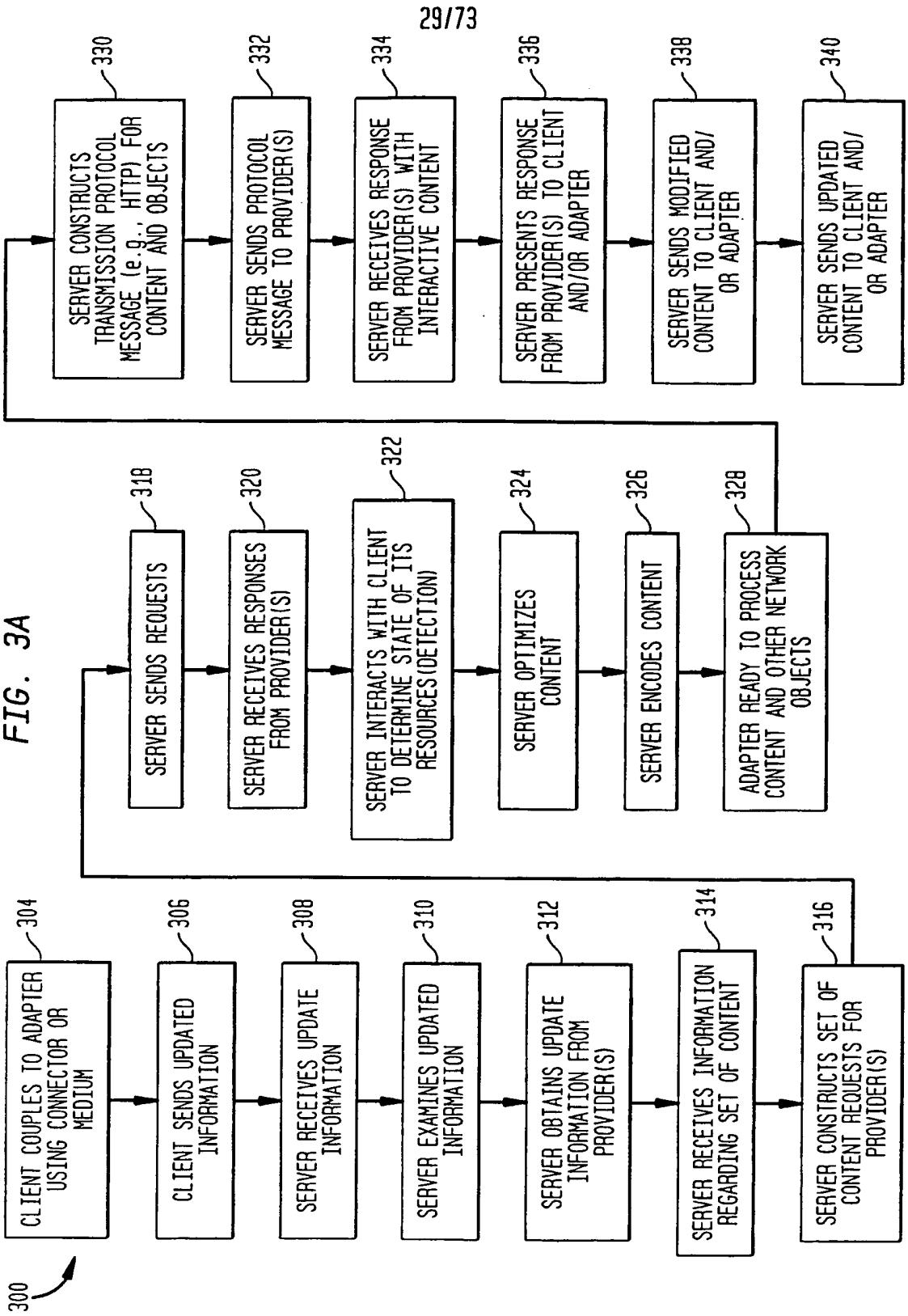

300
CLIENT COUPLES TO ADAPTER USING CONNECTOR OR MEDIUM
304
CLIENT SENDS UPDATED INFORMATION
306
SERVER RECEIVES UPDATE INFORMATION
308
SERVER EXAMINES UPDATED INFORMATION
310
SERVER OBTAINS UPDATE INFORMATION FROM PROVIDER(S)
312
SERVER RECEIVES INFORMATION REGARDING SET OF CONTENT
314
SERVER CONSTRUCTS SET OF CONTENT REQUESTS FOR PROVIDER(S)
316
SERVER SENDS REQUESTS
318
SERVER RECEIVES RESPONSES FROM PROVIDER(S)
320
SERVER INTERACTS WITH CLIENT TO DETERMINE STATE OF ITS RESOURCES(DETECTION)
322
SERVER OPTIMIZES CONTENT
324
SERVER ENCODES CONTENT
326
ADAPTER READY TO PROCESS CONTENT AND OTHER NETWORK OBJECTS
328
SERVER CONSTRUCTS TRANSMISSION PROTOCOL MESSAGE (e.g., HTTP) FOR CONTENT AND OBJECTS
330
SERVER SENDS PROTOCOL MESSAGE TO PROVIDER(S)
332
SERVER RECEIVES RESPONSE FROM PROVIDER(S) WITH INTERACTIVE CONTENT
334
SERVER PRESENTS RESPONSE FROM PROVIDER(S) TO CLIENT AND/OR ADAPTER
336
SERVER SENDS MODIFIED CONTENT TO CLIENT AND/OR ADAPTER
338
SERVER SENDS UPDATED CONTENT TO CLIENT AND/OR ADAPTER
340

FIG. 3B

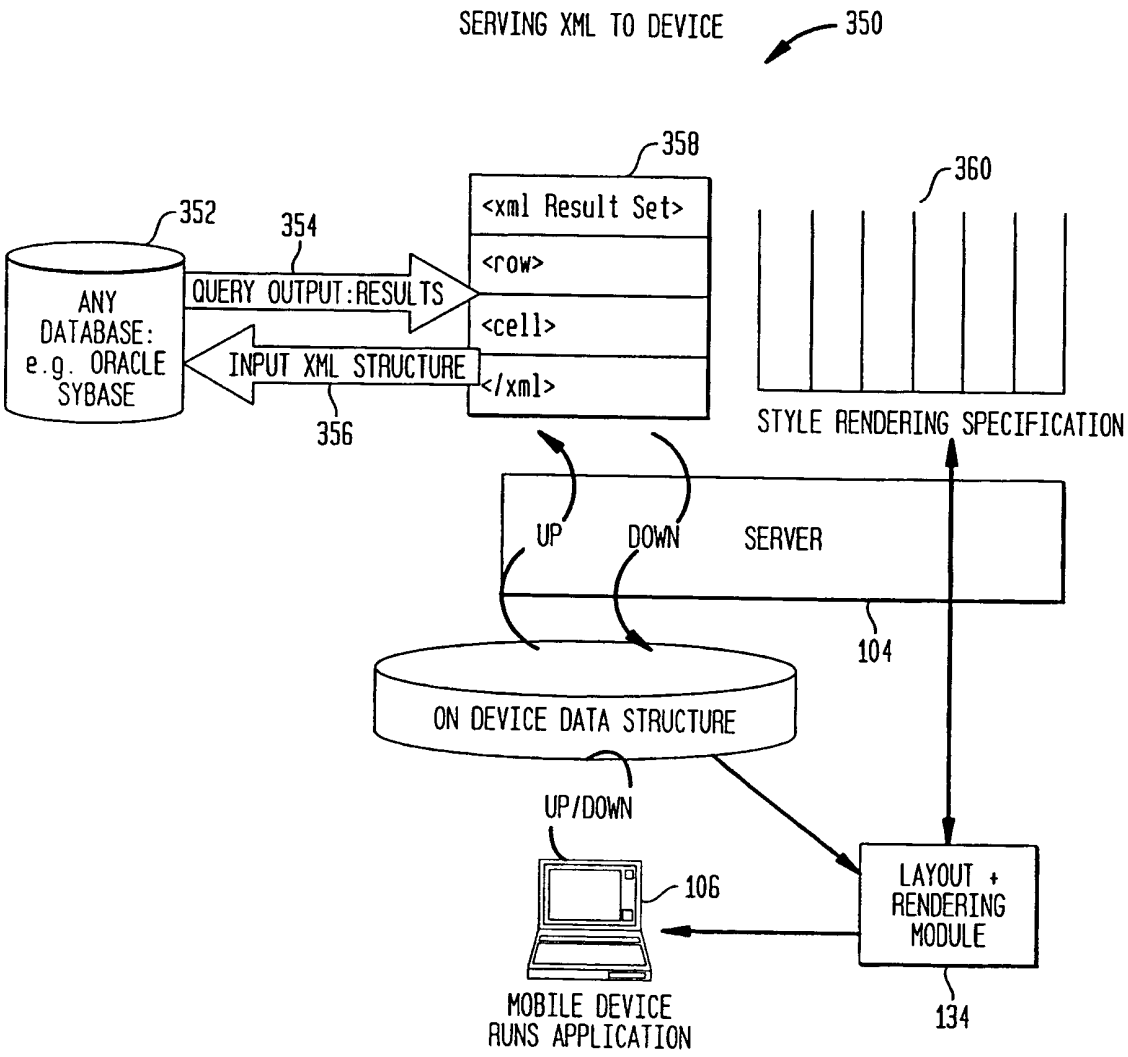
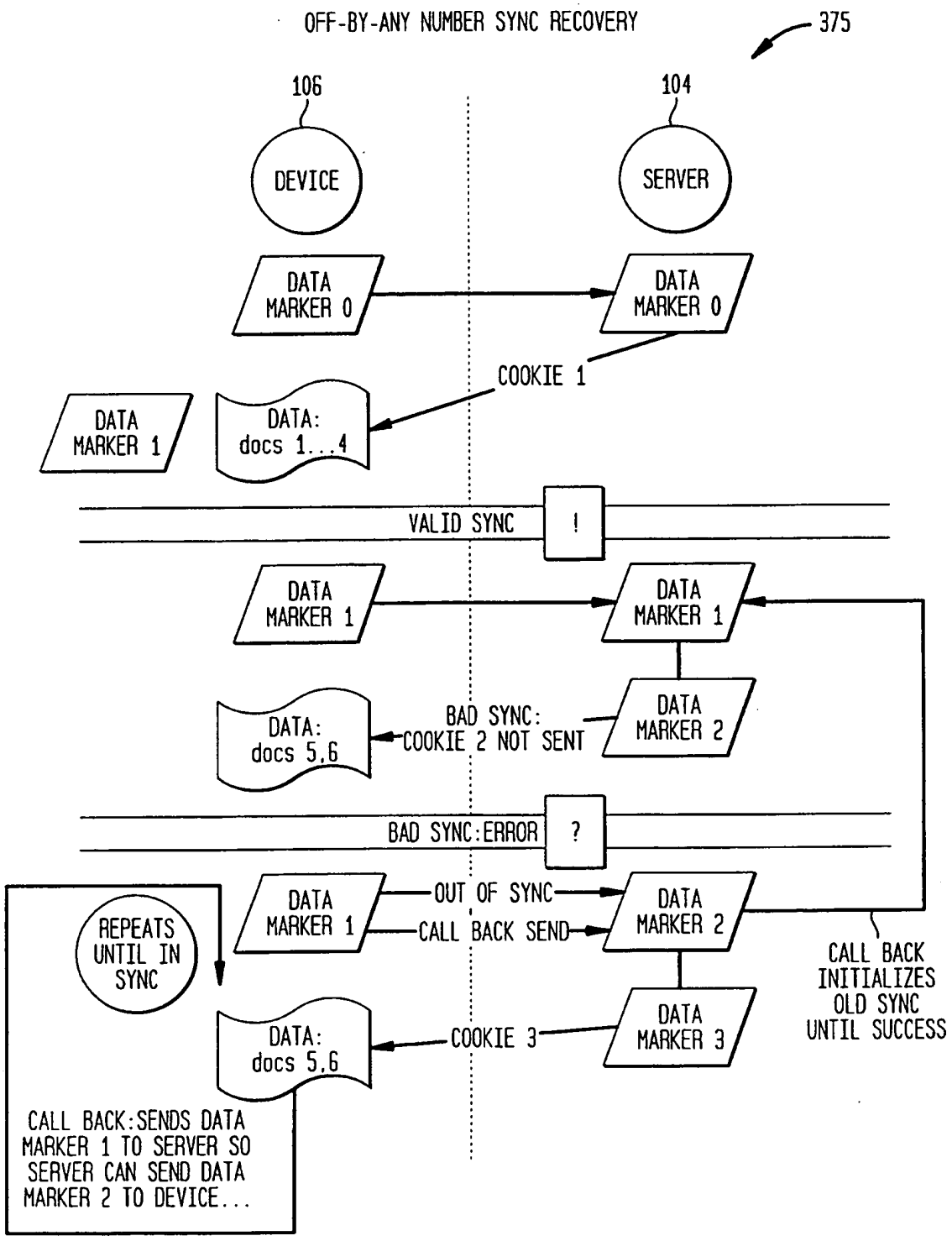

SERVING XML TO DEVICE
350
358
360
352
354
<xml Result Set>
<row>
<cell>
</xml>
ANY DATABASE: e.g. ORACLE SYBASE
QUERY OUTPUT:RESULTS
INPUT XML STRUCTURE
356
STYLE RENDERING SPECIFICATION
UP
DOWN
SERVER
104
ON DEVICE DATA STRUCTURE
UP/DOWN
106
LAYOUT + RENDERING MODULE
MOBILE DEVICE RUNS APPLICATION
134

FIG. 3C


OFF-BY-ANY NUMBER SYNC RECOVERY
375
106
104
DEVICE
SERVER
DATA MARKER 0
DATA MARKER 0
COOKIE 1
DATA MARKER 1
DATA: docs 1...4
VALID SYNC
!
DATA MARKER 1
DATA MARKER 1
DATA: docs 5,6
BAD SYNC: COOKIE 2 NOT SENT
DATA MARKER 2
BAD SYNC:ERROR
?
DATA MARKER 1
OUT OF SYNC
CALL BACK SEND
DATA MARKER 2
CALL BACK INITIALIZES OLD SYNC UNTIL SUCCESS
REPEATS UNTIL IN SYNC
DATA: docs 5,6
COOKIE 3
DATA MARKER 3
CALL BACK:SENDS DATA MARKER 1 TO SERVER SO SERVER CAN SEND DATA MARKER 2 TO DEVICE...

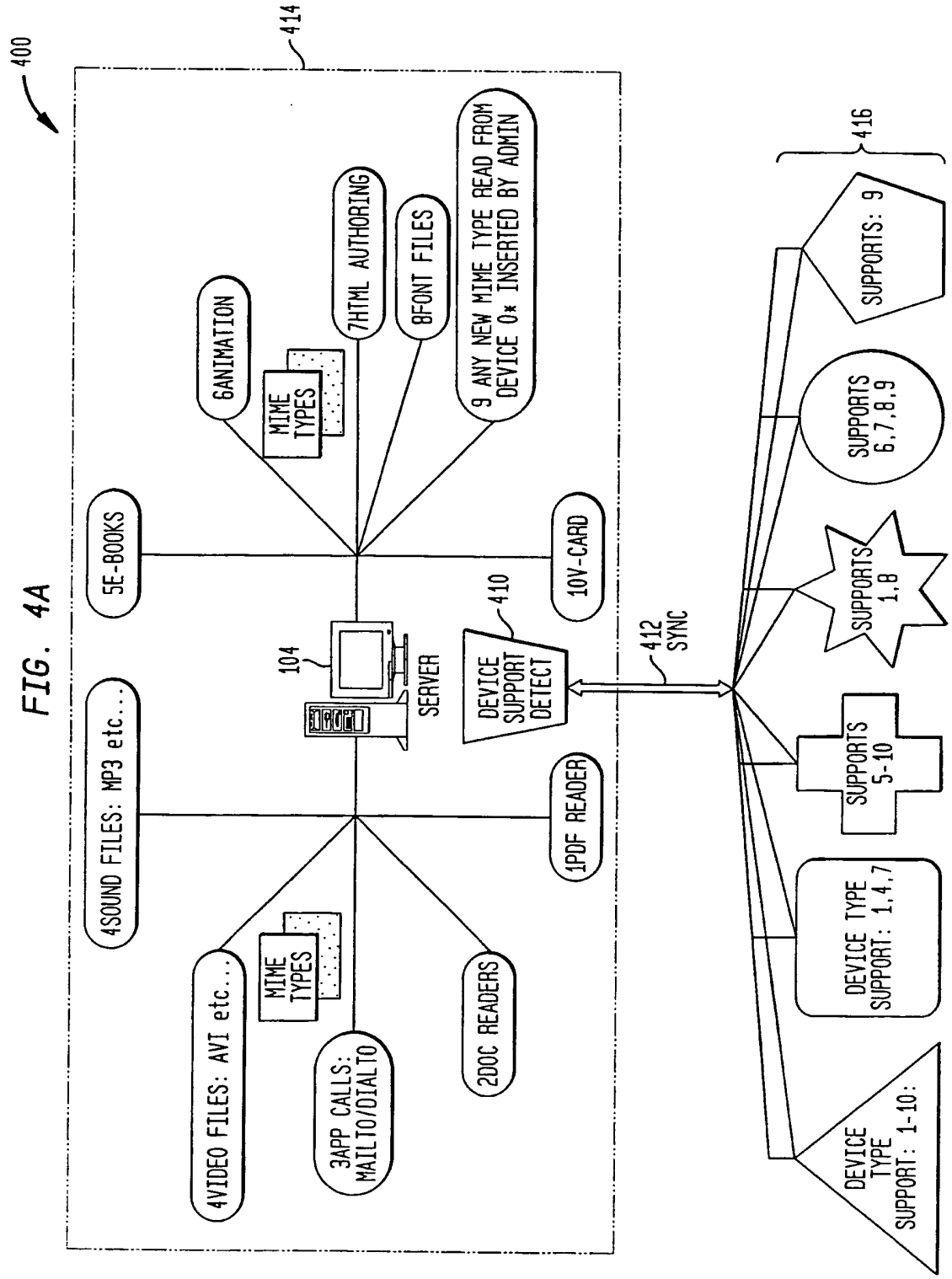

400
414
5E-BOOKS
6ANIMATION
MIME TYPES
7HTML AUTHORING
8FONT FILES
9 ANY NEW MIME TYPE READ FROM DEVICE 0x INSERTED BY ADMIN
10V-CARD
4SOUND FILES: MP3 etc...
4VIDEO FILES: AVI etc...
MIME TYPES
3APP CALLS: MAILTO/DIALTO
2DOC READERS
1PDF READER
104
SERVER
410
DEVICE SUPPORT DETECT
412
SYNC
416
SUPPORTS: 9
SUPPORTS 6,7,8,9
SUPPORTS 1,8
SUPPORTS 5-10
DEVICE TYPE SUPPORT: 1,4,7
DEVICE TYPE SUPPORT: 1-10:

FIG. 4A

FIG. 4B

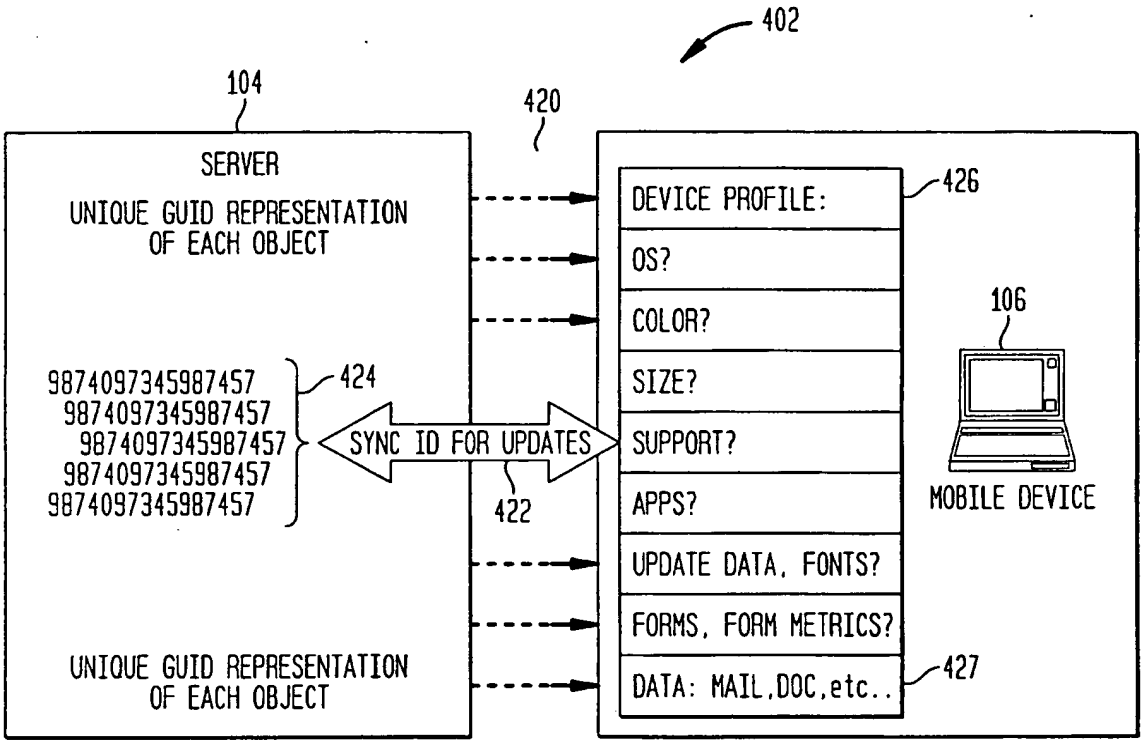

402
104
420
SERVER
UNIQUE GUID REPRESENTATION OF EACH OBJECT
9874097345987457
9874097345987457
9874097345987457
9874097345987457
9874097345987457
424
SYNC ID FOR UPDATES
422
UNIQUE GUID REPRESENTATION OF EACH OBJECT
DEVICE PROFILE:
OS?
COLOR?
SIZE?
SUPPORT?
APPS?
UPDATE DATA, FONTS?
FORMS, FORM METRICS?
DATA: MAIL,DOC,etc..
426
427
106
MOBILE DEVICE

FIG. 5A

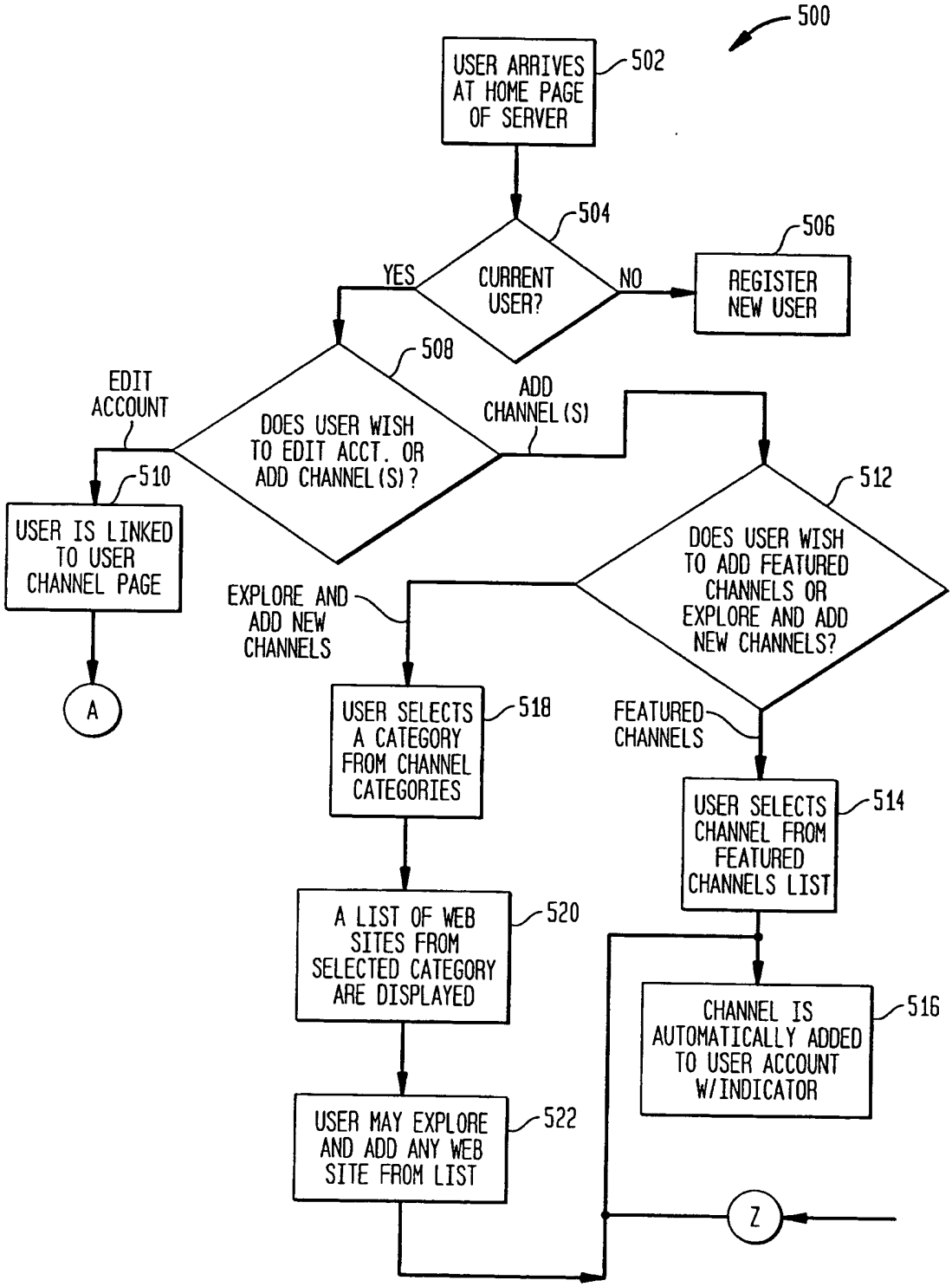

500
USER ARRIVES AT HOME PAGE OF SERVER
502
CURRENT USER?
504
YES
NO
REGISTER NEW USER
506
DOES USER WISH TO EDIT ACCT. OR ADD CHANNEL(S)?
508
EDIT ACCOUNT
ADD CHANNEL(S)
USER IS LINKED TO USER CHANNEL PAGE
510
A
DOES USER WISH TO ADD FEATURED CHANNELS OR EXPLORE AND ADD NEW CHANNELS?
512
EXPLORE AND ADD NEW CHANNELS
FEATURED CHANNELS
USER SELECTS A CATEGORY FROM CHANNEL CATEGORIES
518
USER SELECTS CHANNEL FROM FEATURED CHANNELS LIST
514
A LIST OF WEB SITES FROM SELECTED CATEGORY ARE DISPLAYED
520
CHANNEL IS AUTOMATICALLY ADDED TO USER ACCOUNT W/INDICATOR
516
USER MAY EXPLORE AND ADD ANY WEB SITE FROM LIST
522
Z

FIG. 5B

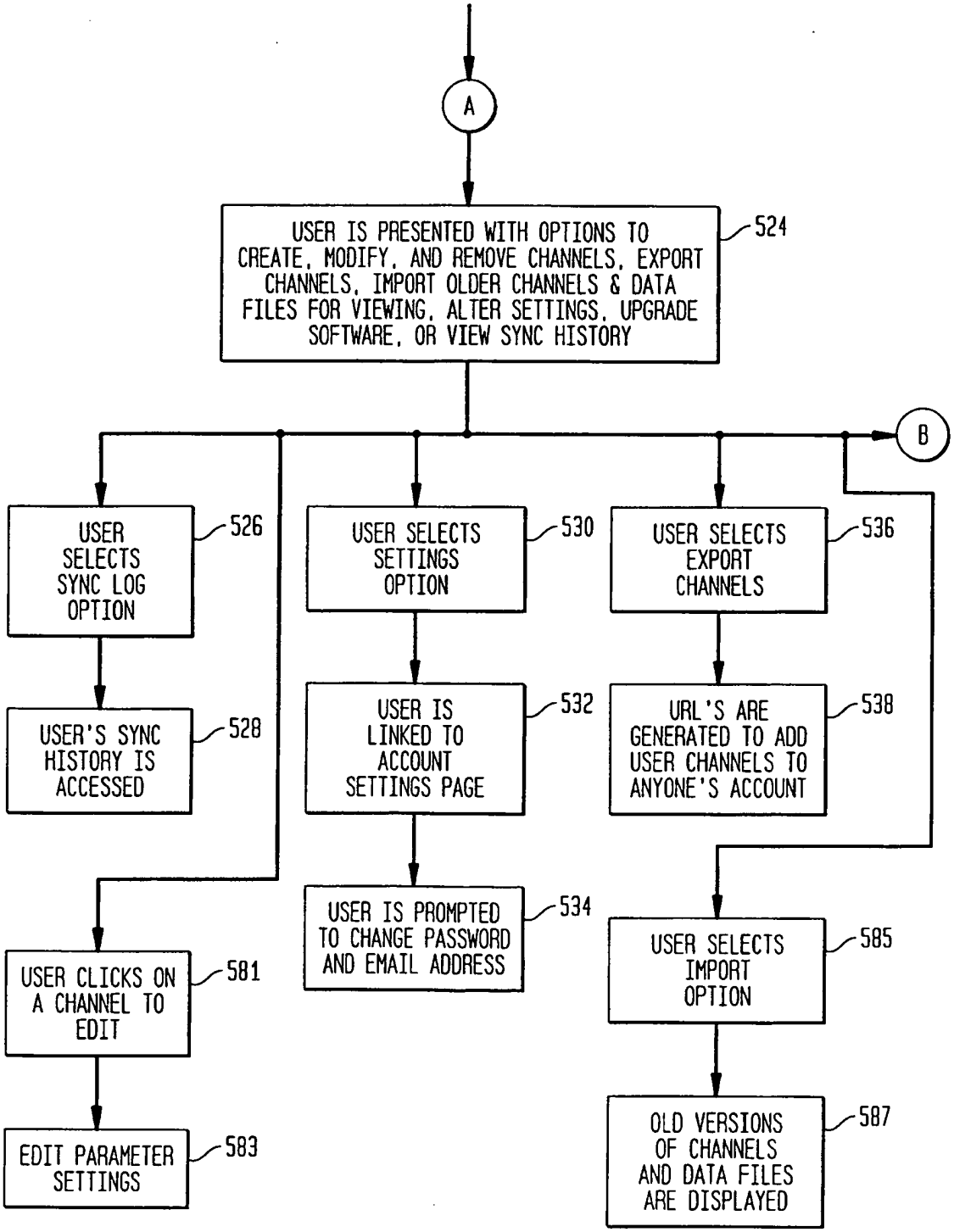

A
USER IS PRESENTED WITH OPTIONS TO CREATE, MODIFY, AND REMOVE CHANNELS, EXPORT CHANNELS, IMPORT OLDER CHANNELS & DATA FILES FOR VIEWING, ALTER SETTINGS, UPGRADE SOFTWARE, OR VIEW SYNC HISTORY
524
B
USER SELECTS SYNC LOG OPTION
526
USER'S SYNC HISTORY IS ACCESSED
528
USER CLICKS ON A CHANNEL TO EDIT
581
EDIT PARAMETER SETTINGS
583
USER SELECTS SETTINGS OPTION
530
USER IS LINKED TO ACCOUNT SETTINGS PAGE
532
USER IS PROMPTED TO CHANGE PASSWORD AND EMAIL ADDRESS
534
USER SELECTS EXPORT CHANNELS
536
URL'S ARE GENERATED TO ADD USER CHANNELS TO ANYONE'S ACCOUNT
538
USER SELECTS IMPORT OPTION
585
OLD VERSIONS OF CHANNELS AND DATA FILES ARE DISPLAYED
587

FIG. 5C

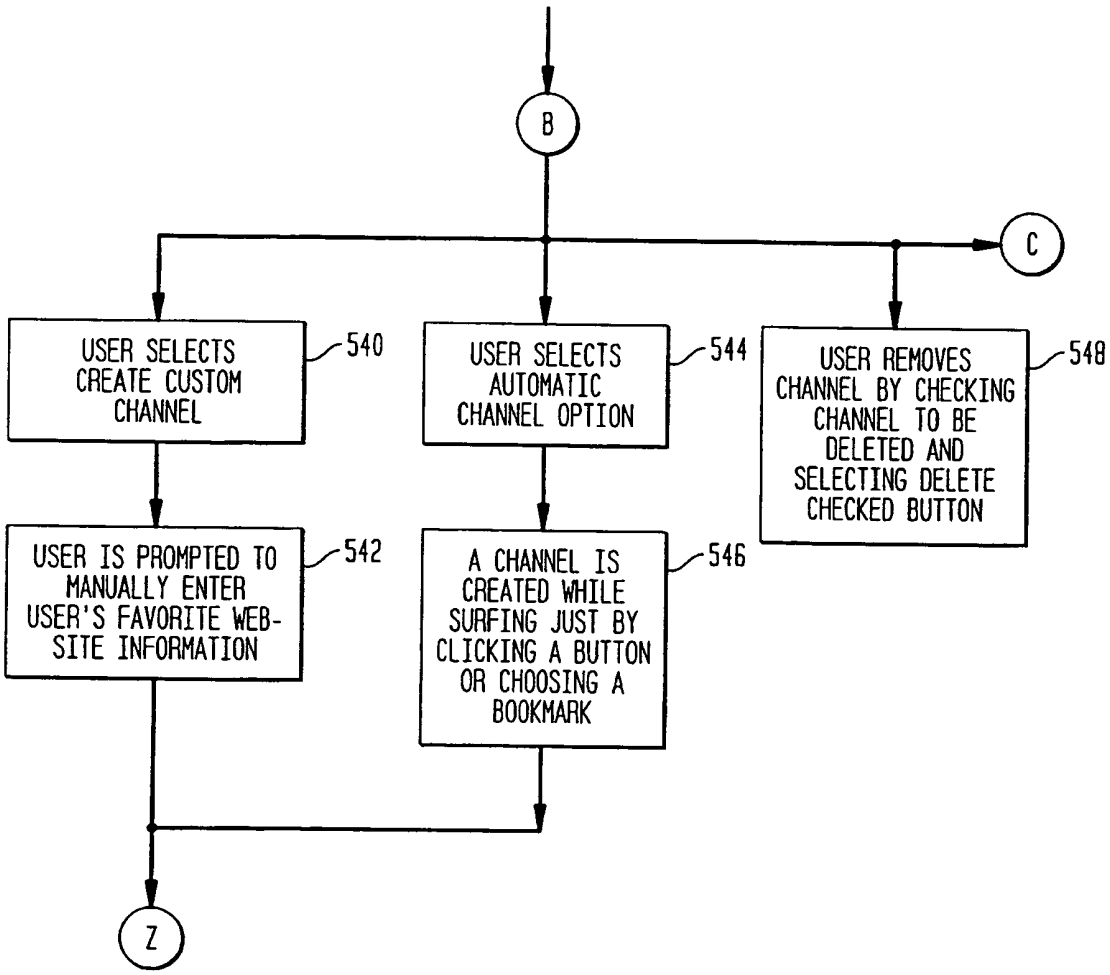

B
C
USER SELECTS CREATE CUSTOM CHANNEL
540
USER IS PROMPTED TO MANUALLY ENTER USER'S FAVORITE WEB-SITE INFORMATION
542
USER SELECTS AUTOMATIC CHANNEL OPTION
544
A CHANNEL IS CREATED WHILE SURFING JUST BY CLICKING A BUTTON OR CHOOSING A BOOKMARK
546
USER REMOVES CHANNEL BY CHECKING CHANNEL TO BE DELETED AND SELECTING DELETE CHECKED BUTTON
548
Z

FIG. 5D

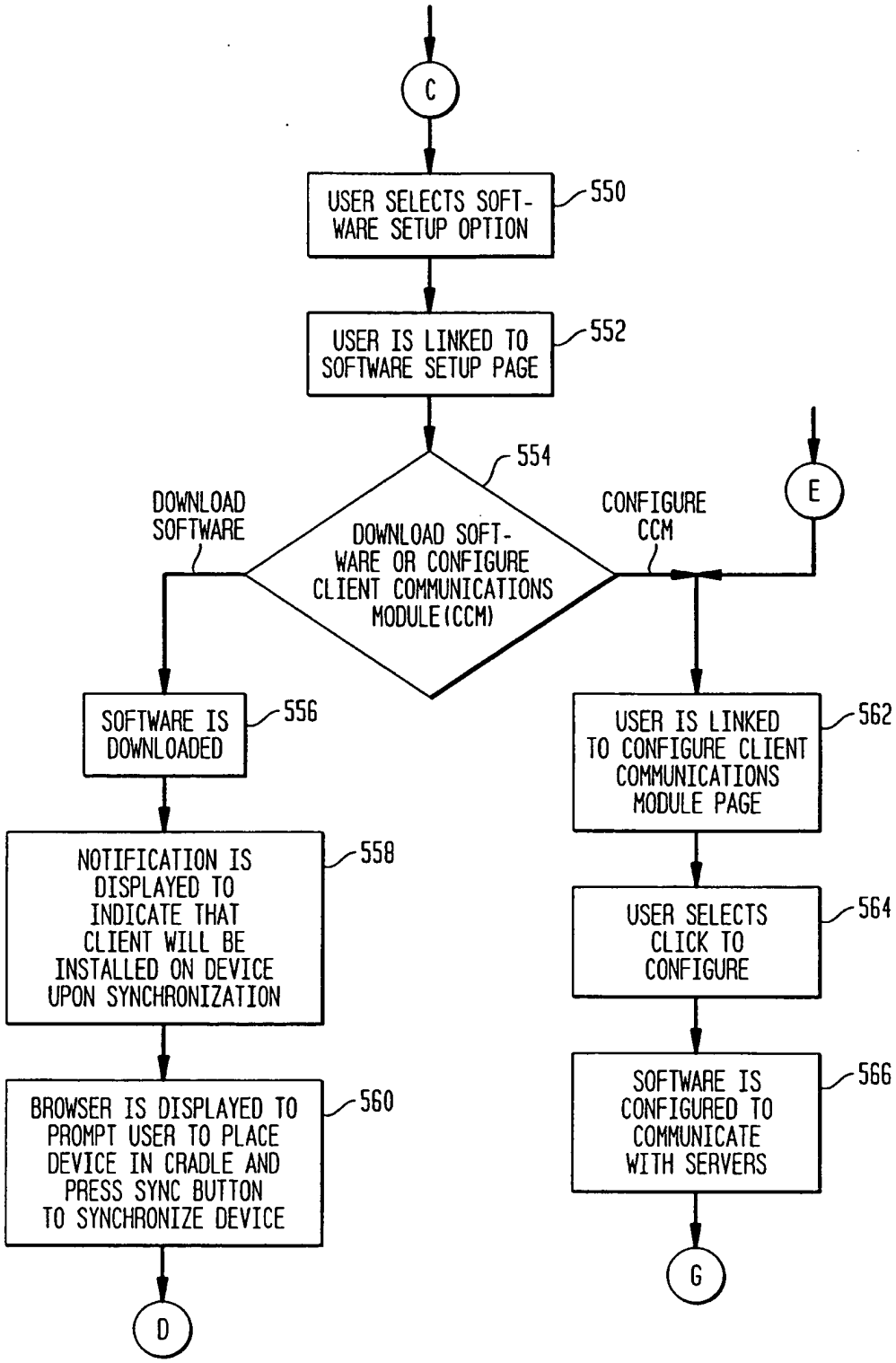

C
USER SELECTS SOFTWARE SETUP OPTION
550
USER IS LINKED TO SOFTWARE SETUP PAGE
552
554
DOWNLOAD SOFTWARE
DOWNLOAD SOFTWARE OR CONFIGURE CLIENT COMMUNICATIONS MODULE(CCM)
CONFIGURE CCM
E
SOFTWARE IS DOWNLOADED
556
NOTIFICATION IS DISPLAYED TO INDICATE THAT CLIENT WILL BE INSTALLED ON DEVICE UPON SYNCHRONIZATION
558
BROWSER IS DISPLAYED TO PROMPT USER TO PLACE DEVICE IN CRADLE AND PRESS SYNC BUTTON TO SYNCHRONIZE DEVICE
560
D
USER IS LINKED TO CONFIGURE CLIENT COMMUNICATIONS MODULE PAGE
562
USER SELECTS CLICK TO CONFIGURE
564
SOFTWARE IS CONFIGURED TO COMMUNICATE WITH SERVERS
566
G

FIG. 5E

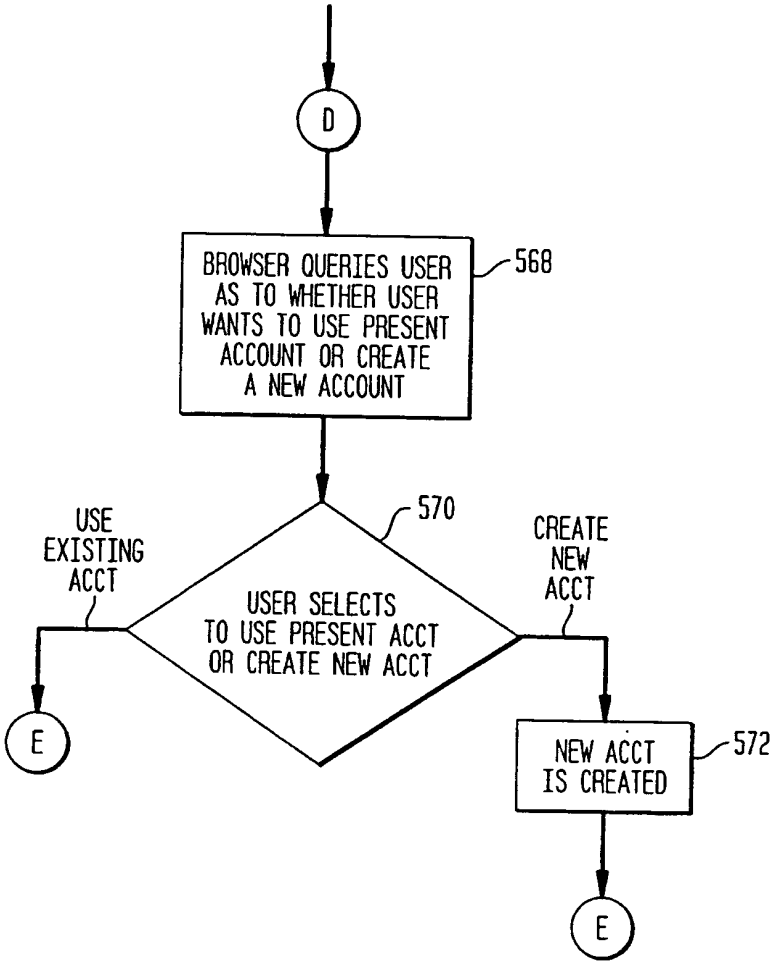

D
BROWSER QUERIES USER AS TO WHETHER USER WANTS TO USE PRESENT ACCOUNT OR CREATE A NEW ACCOUNT
568
570
USE EXISTING ACCT
USER SELECTS TO USE PRESENT ACCT OR CREATE NEW ACCT
CREATE NEW ACCT
E
NEW ACCT IS CREATED
572
E

FIG. 5F

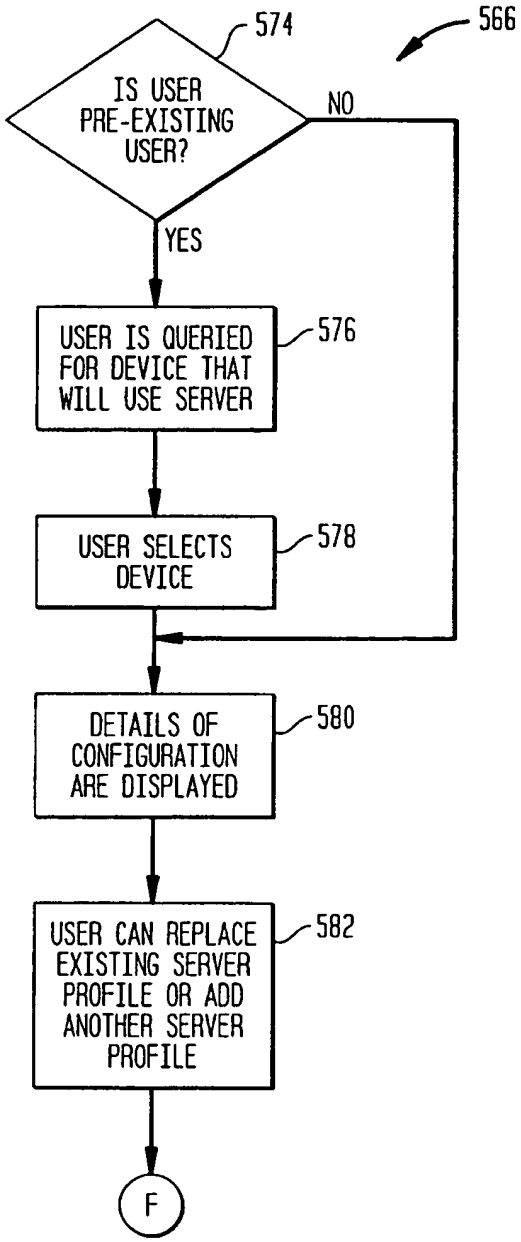

574
566
IS USER PRE-EXISTING USER?
NO
YES
USER IS QUERIED FOR DEVICE THAT WILL USE SERVER
576
USER SELECTS DEVICE
578
DETAILS OF CONFIGURATION ARE DISPLAYED
580
USER CAN REPLACE EXISTING SERVER PROFILE OR ADD ANOTHER SERVER PROFILE
582
F

FIG. 5G

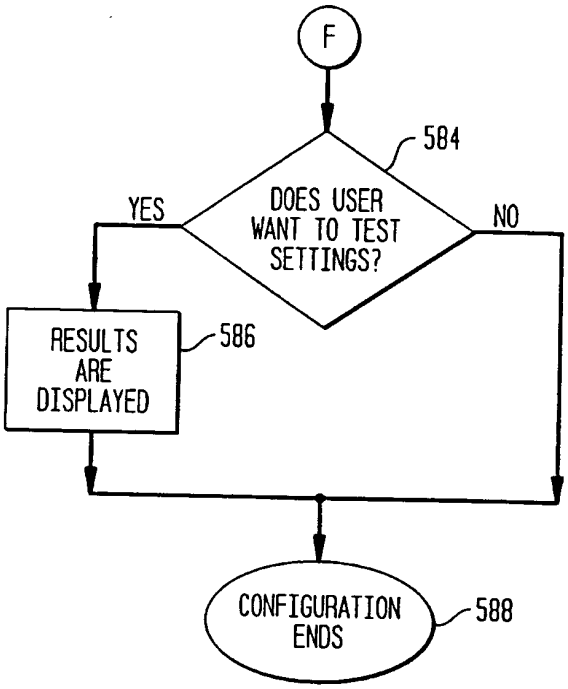

F
584
DOES USER WANT TO TEST SETTINGS?
YES
NO
RESULTS ARE DISPLAYED
586
CONFIGURATION ENDS
588

FIG. 5H

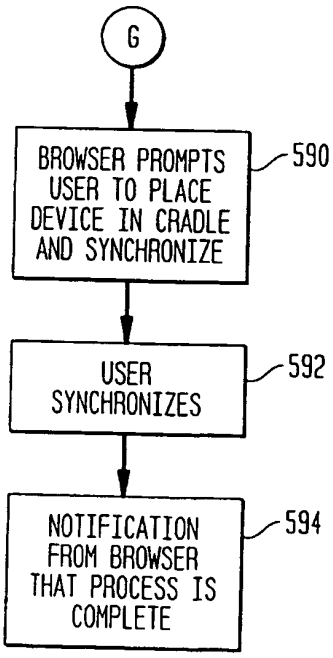

G
BROWSER PROMPTS USER TO PLACE DEVICE IN CRADLE AND SYNCHRONIZE
590
USER SYNCHRONIZES
592
NOTIFICATION FROM BROWSER THAT PROCESS IS COMPLETE
594

FIG. 5I

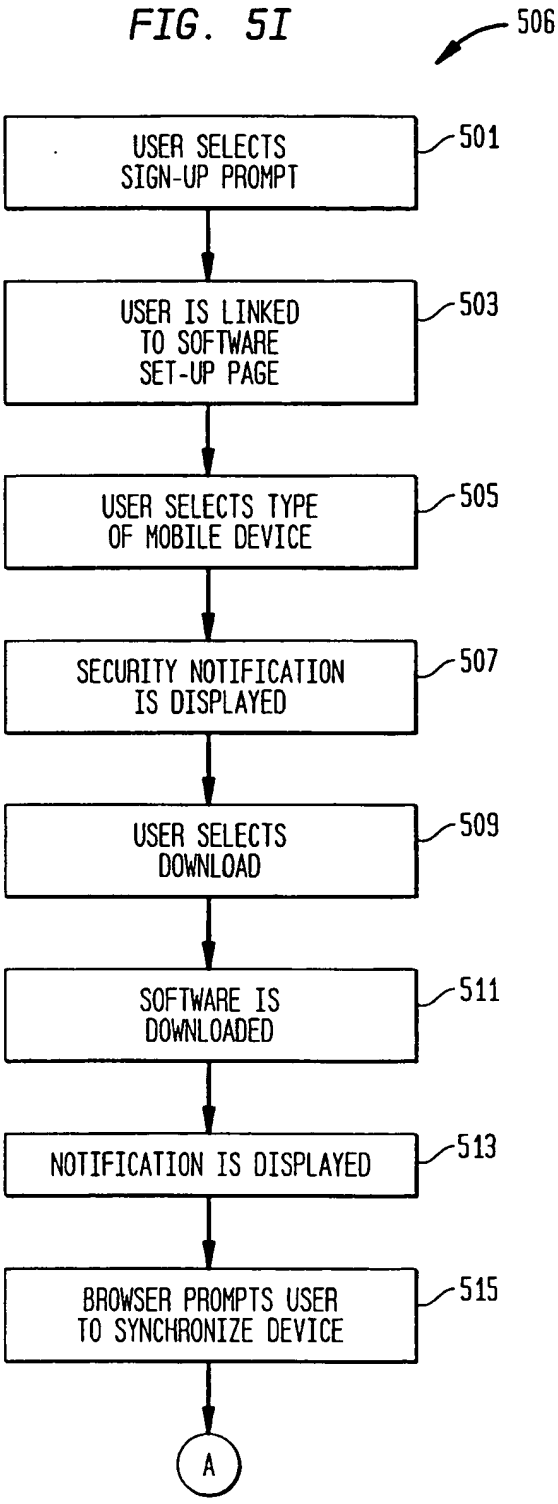

506
USER SELECTS SIGN-UP PROMPT
501
USER IS LINKED TO SOFTWARE SET-UP PAGE
503
USER SELECTS TYPE OF MOBILE DEVICE
505
SECURITY NOTIFICATION IS DISPLAYED
507
USER SELECTS DOWNLOAD
509
SOFTWARE IS DOWNLOADED
511
NOTIFICATION IS DISPLAYED
513
BROWSER PROMPTS USER TO SYNCHRONIZE DEVICE
515
A

FIG. 5J

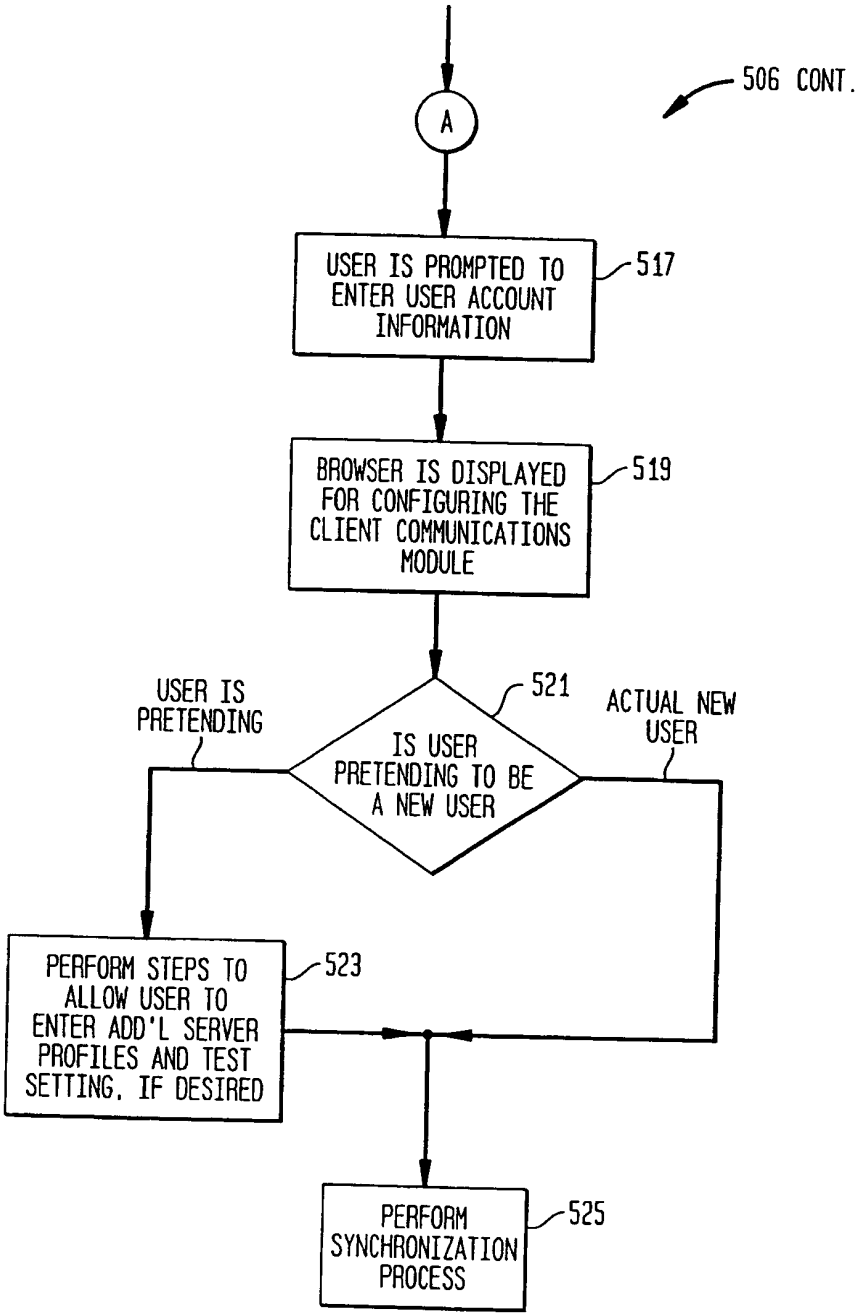

506 CONT.
A
USER IS PROMPTED TO ENTER USER ACCOUNT INFORMATION
517
BROWSER IS DISPLAYED FOR CONFIGURING THE CLIENT COMMUNICATIONS MODULE
519
521
IS USER PRETENDING TO BE A NEW USER
USER IS PRETENDING
ACTUAL NEW USER
PERFORM STEPS TO ALLOW USER TO ENTER ADD'L SERVER PROFILES AND TEST SETTING, IF DESIRED
523
PERFORM SYNCHRONIZATION PROCESS
525

FIG. 5K

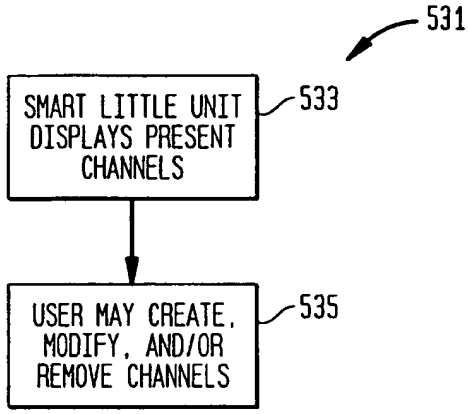

531
SMART LITTLE UNIT DISPLAYS PRESENT CHANNELS
533
USER MAY CREATE, MODIFY, AND/OR REMOVE CHANNELS
535

FIG. 5L

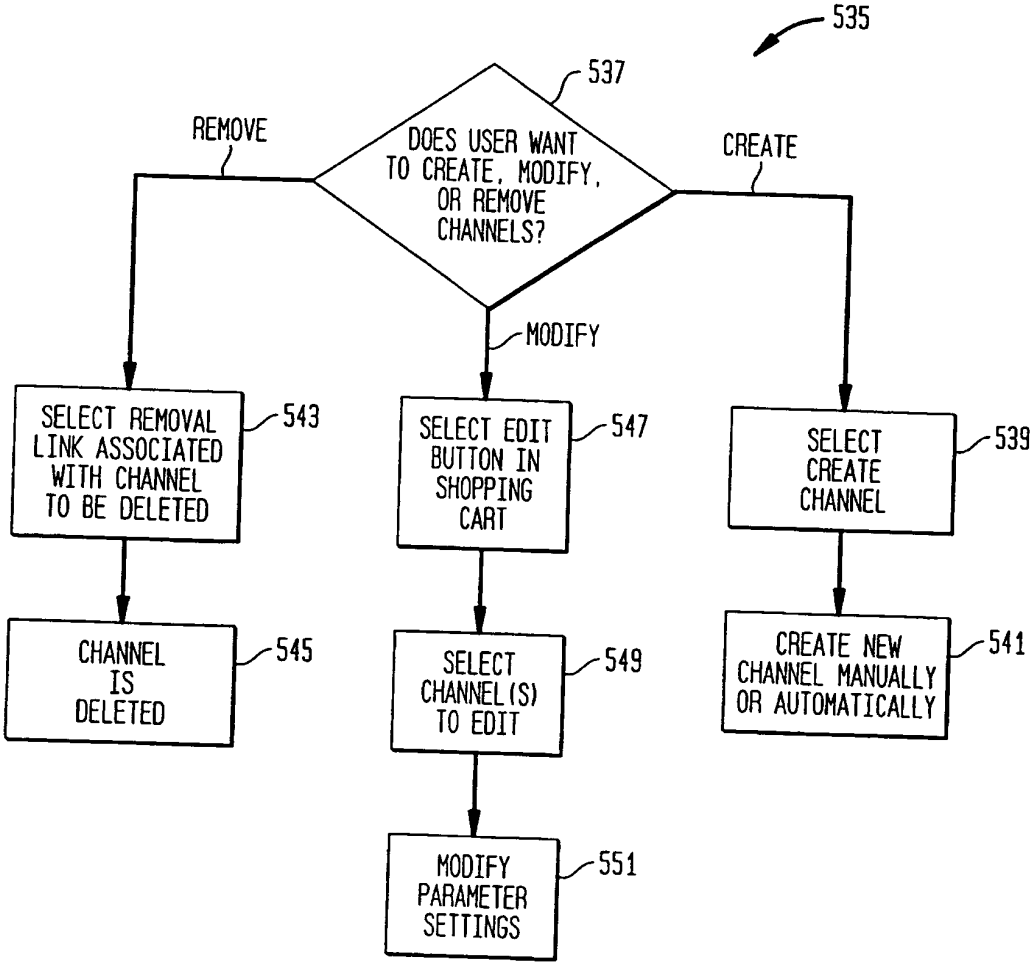

535
537
DOES USER WANT TO CREATE, MODIFY, OR REMOVE CHANNELS?
REMOVE
MODIFY
CREATE
SELECT REMOVAL LINK ASSOCIATED WITH CHANNEL TO BE DELETED
543
CHANNEL IS DELETED
545
SELECT EDIT BUTTON IN SHOPPING CART
547
SELECT CHANNEL(S) TO EDIT
549
MODIFY PARAMETER SETTINGS
551
SELECT CREATE CHANNEL
539
CREATE NEW CHANNEL MANUALLY OR AUTOMATICALLY
541

FIG. 5M

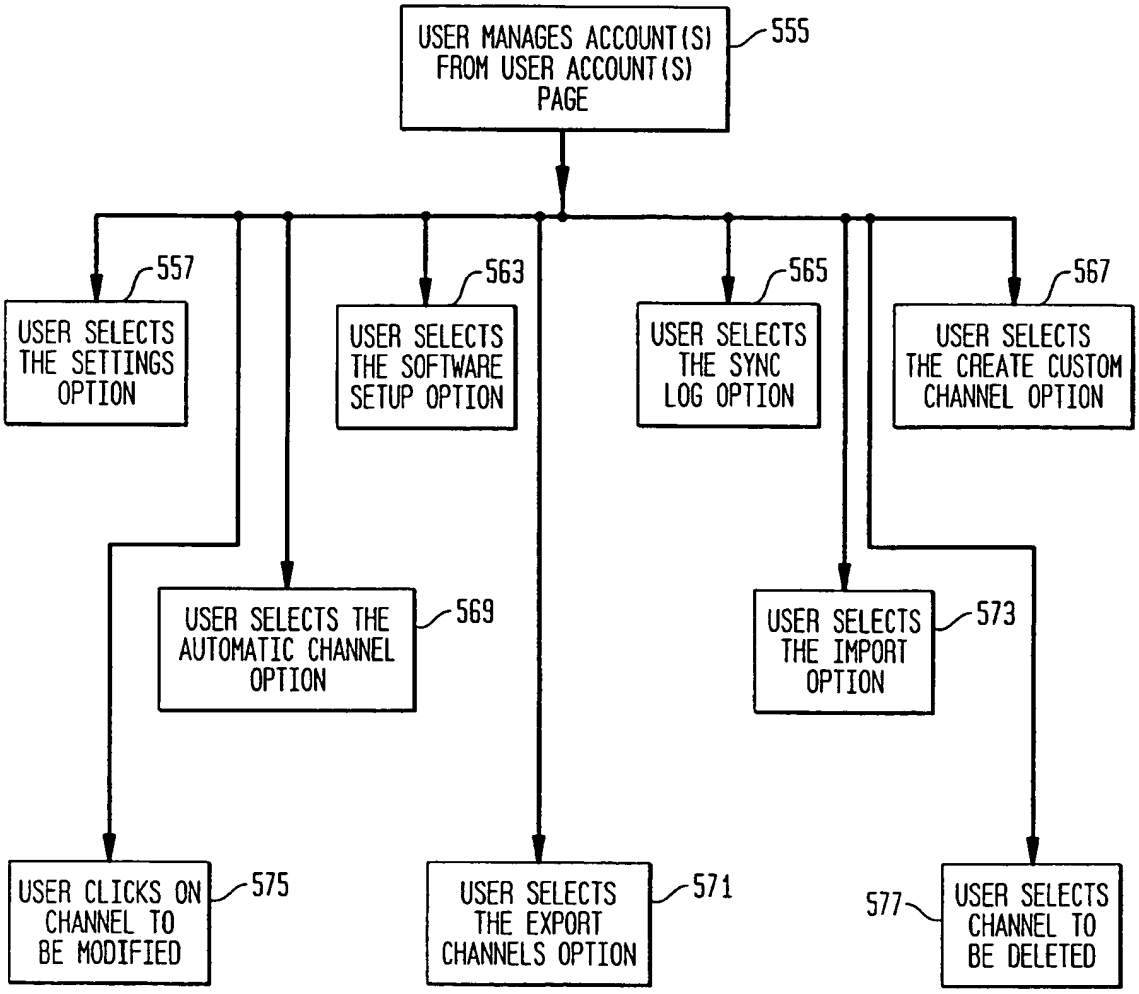
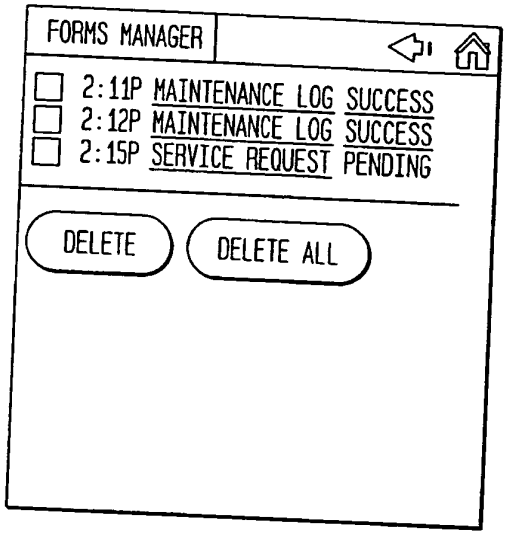

USER MANAGES ACCOUNT(S) FROM USER ACCOUNT(S) PAGE
555
USER SELECTS THE SETTINGS OPTION
557
USER SELECTS THE SOFTWARE SETUP OPTION
563
USER SELECTS THE SYNC LOG OPTION
565
USER SELECTS THE CREATE CUSTOM CHANNEL OPTION
567
USER SELECTS THE AUTOMATIC CHANNEL OPTION
569
USER SELECTS THE IMPORT OPTION
573
USER CLICKS ON CHANNEL TO BE MODIFIED
575
USER SELECTS THE EXPORT CHANNELS OPTION
571
USER SELECTS CHANNEL TO BE DELETED
577

FIG. 6


FORMS MANAGER
2:11P MAINTENANCE LOG SUCCESS
2:12P MAINTENANCE LOG SUCCESS
2:15P SERVICE REQUEST PENDING
DELETE
DELETE ALL

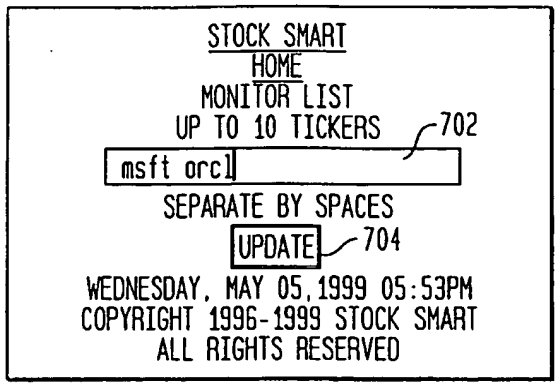


FIG. 7

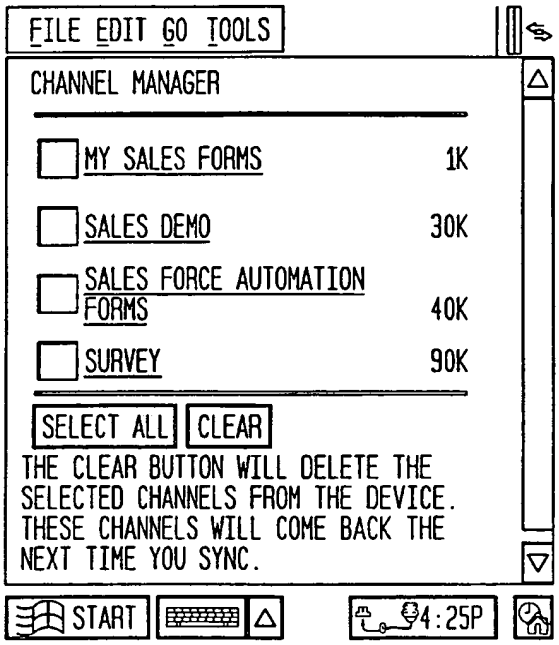


FIG. 8

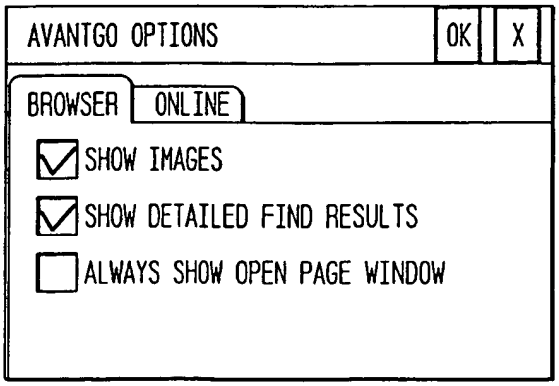


FIG. 9

FIG. 10

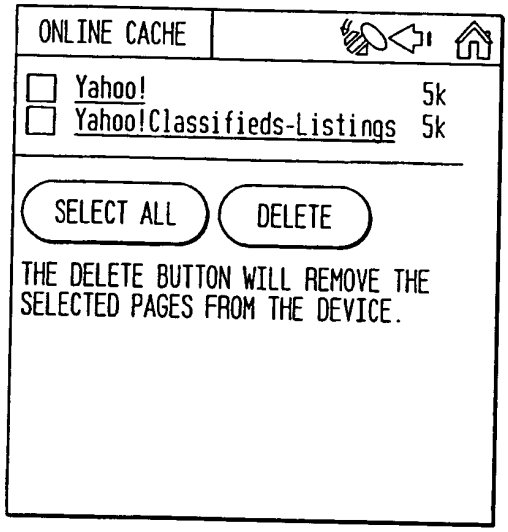

ONLINE CACHE
Yahoo! 5k
Yahoo!Classifieds-Listings 5k
SELECT ALL
DELETE
THE DELETE BUTTON WILL REMOVE THE SELECTED PAGES FROM THE DEVICE.

FIG. 11

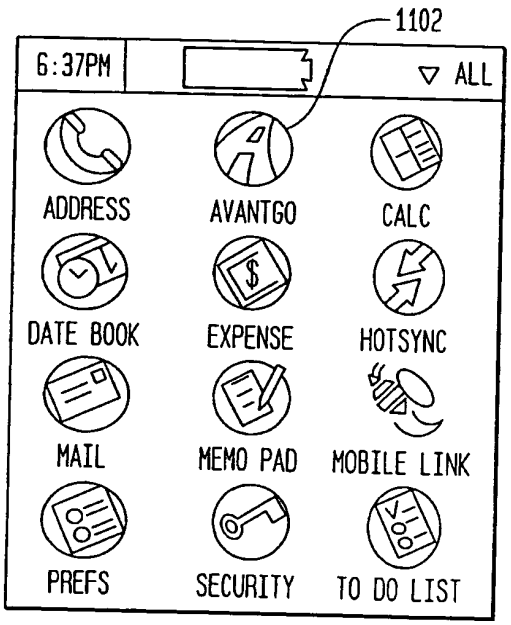

1102
6:37PM
ALL
ADDRESS
AVANTGO
CALC
DATE BOOK
EXPENSE
HOTSYNC
MAIL
MEMO PAD
MOBILE LINK
PREFS
SECURITY
TO DO LIST

FIG. 12



FIG. 13

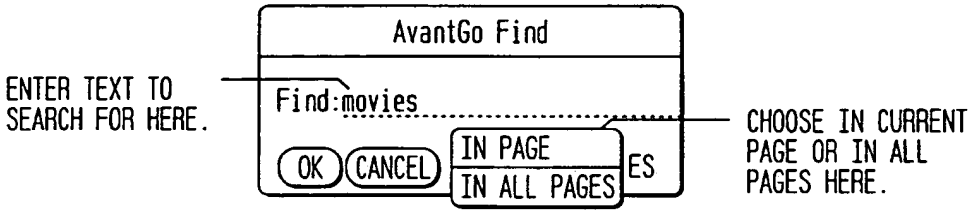


FIG. 14

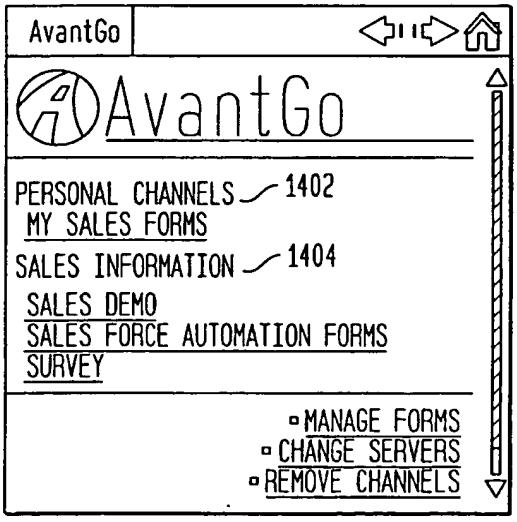


FIG. 15

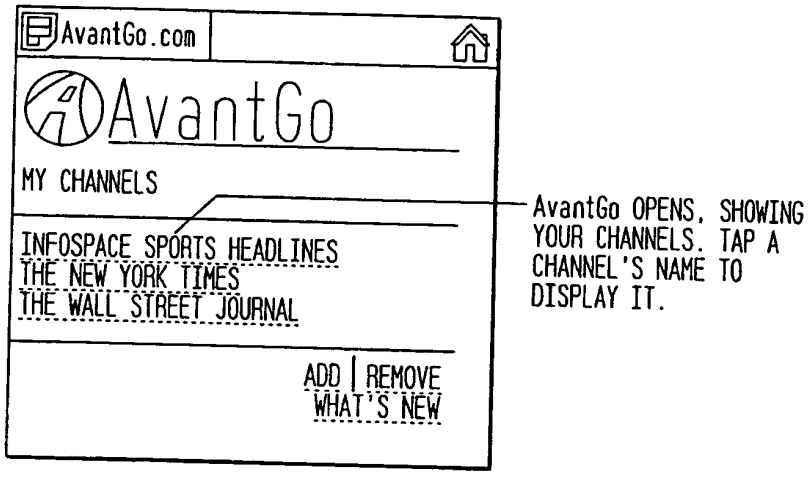


FIG. 16

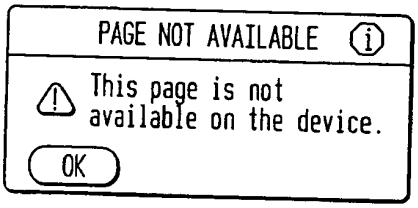


FIG. 17

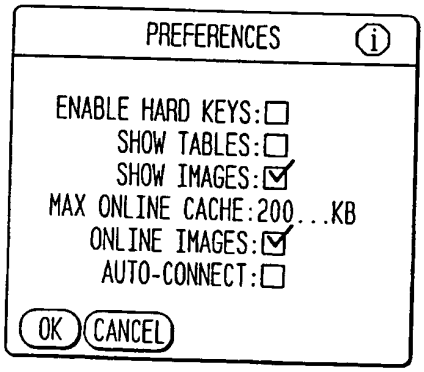


FIG. 18

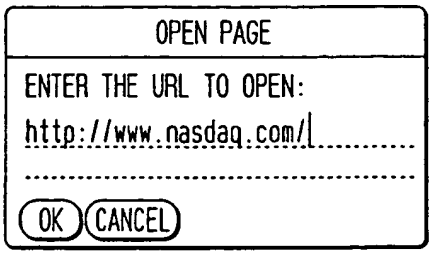


FIG. 19

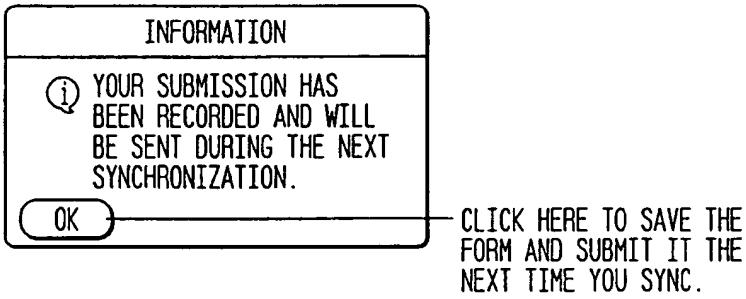


FIG. 20

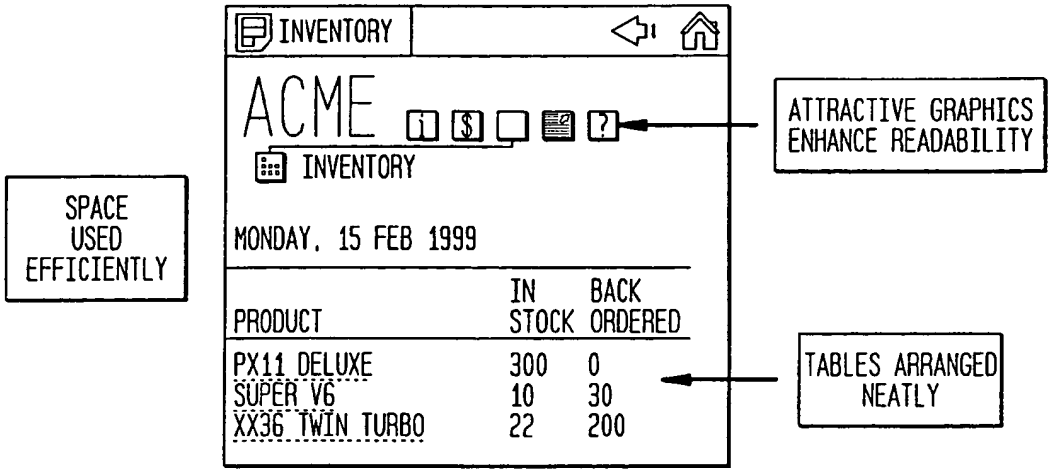


FIG. 21



INCREASE FONT

FIG. 22



DECREASE FONT

FIG. 23

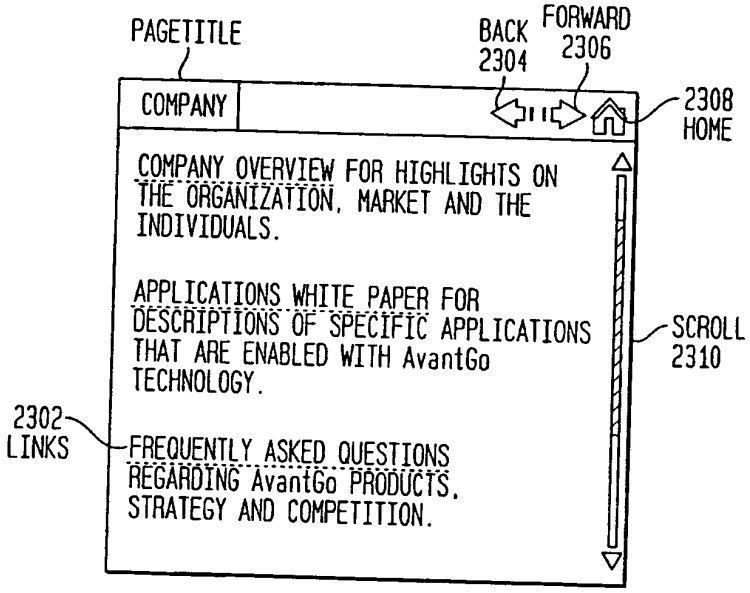


FIG. 24

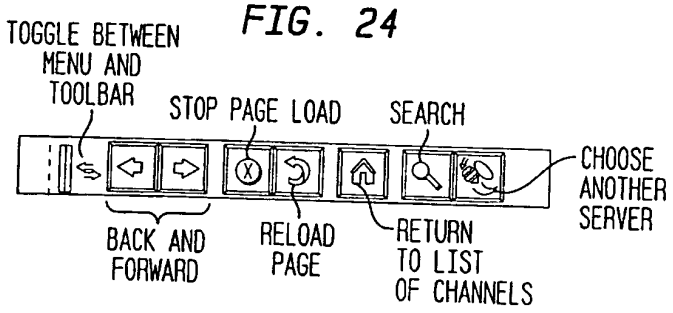


FIG. 25


AvantGo
MY CHANNELS
CITY SNAPSHOTS
FOX SPORTS
MARKETING CENTRAL
THE WALL STREET JOURNAL
WIRED NEWS
ADD | REMOVE
START
7:22P

FIG. 26

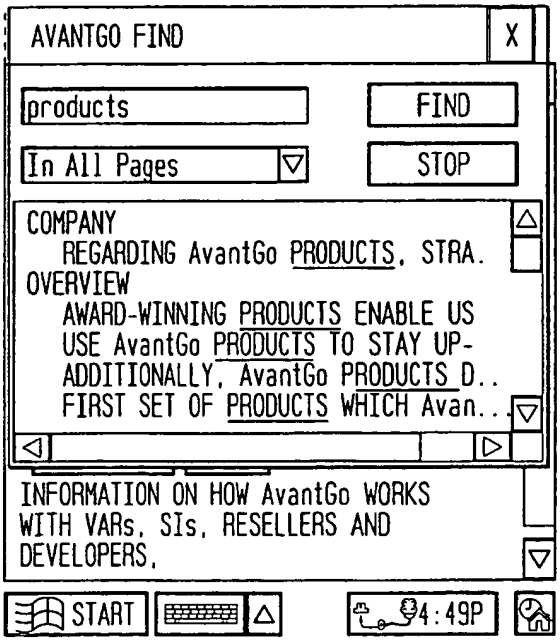

AVANTGO FIND
X
products
FIND
In All Pages
STOP
COMPANY
REGARDING AvantGo PRODUCTS, STRA.
OVERVIEW
AWARD-WINNING PRODUCTS ENABLE US
USE AvantGo PRODUCTS TO STAY UP-
ADDITIONALLY, AvantGo PRODUCTS D..
FIRST SET OF PRODUCTS WHICH Avan..
INFORMATION ON HOW AvantGo WORKS
WITH VARs, SIs, RESELLERS AND
DEVELOPERS,
START
4:49P

FIG. 27

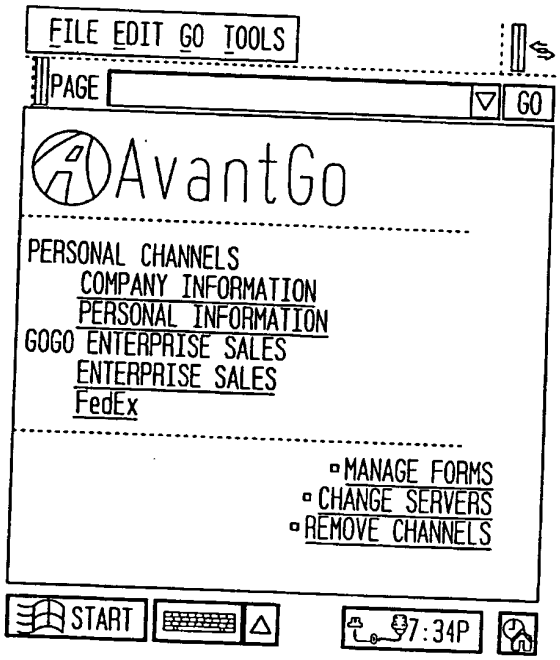

FILE EDIT GO TOOLS
PAGE
GO
AvantGo
PERSONAL CHANNELS
COMPANY INFORMATION
PERSONAL INFORMATION
GOGO ENTERPRISE SALES
ENTERPRISE SALES
FedEx
MANAGE FORMS
CHANGE SERVERS
REMOVE CHANNELS
START
7:34P

FIG. 28

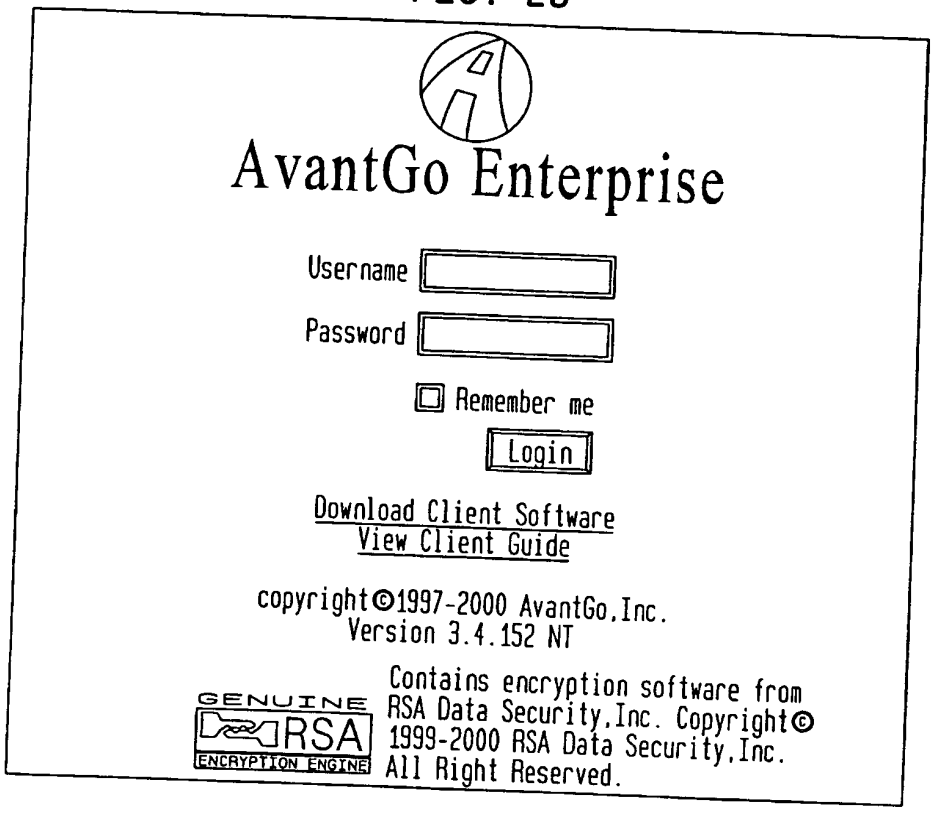

AvantGo Enterprise
Username
Password
Remember me
Login
Download Client Software
View Client Guide
copyright©1997-2000 AvantGo,Inc.
Version 3.4.152 NT
GENUINE RSA ENCRYPTION ENGINE
Contains encryption software from RSA Data Security,Inc. Copyright© 1999-2000 RSA Data Security,Inc. All Right Reserved.

FIG. 29

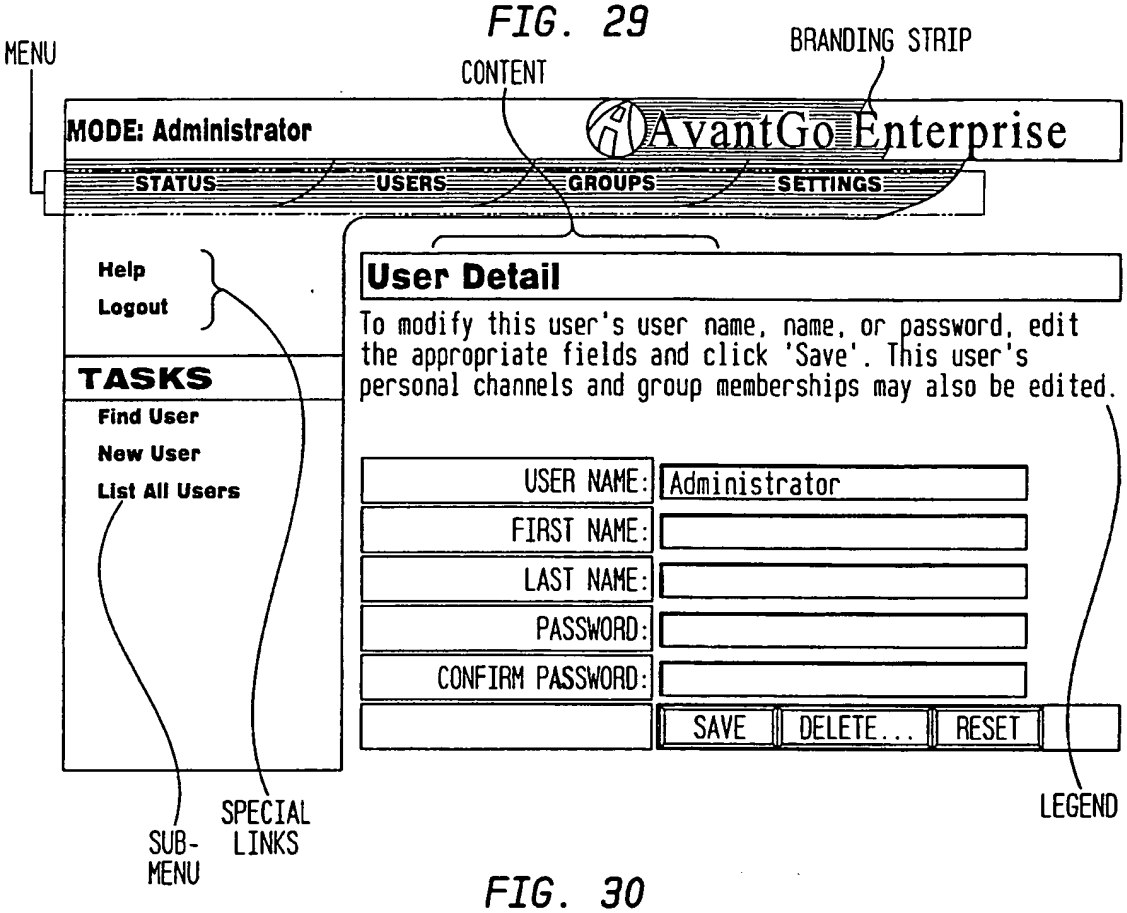

MENU
CONTENT
BRANDING STRIP
MODE: Administrator
AvantGo Enterprise
STATUS
USERS
GROUPS
SETTINGS
Help
Logout
TASKS
Find User
New User
List All Users
User Detail
To modify this user's user name, name, or password, edit the appropriate fields and click 'Save'. This user's personal channels and group memberships may also be edited.
USER NAME:
Administrator
FIRST NAME:
LAST NAME:
PASSWORD:
CONFIRM PASSWORD:
SAVE
DELETE...
RESET
SUB-MENU
SPECIAL LINKS
LEGEND

FIG. 30

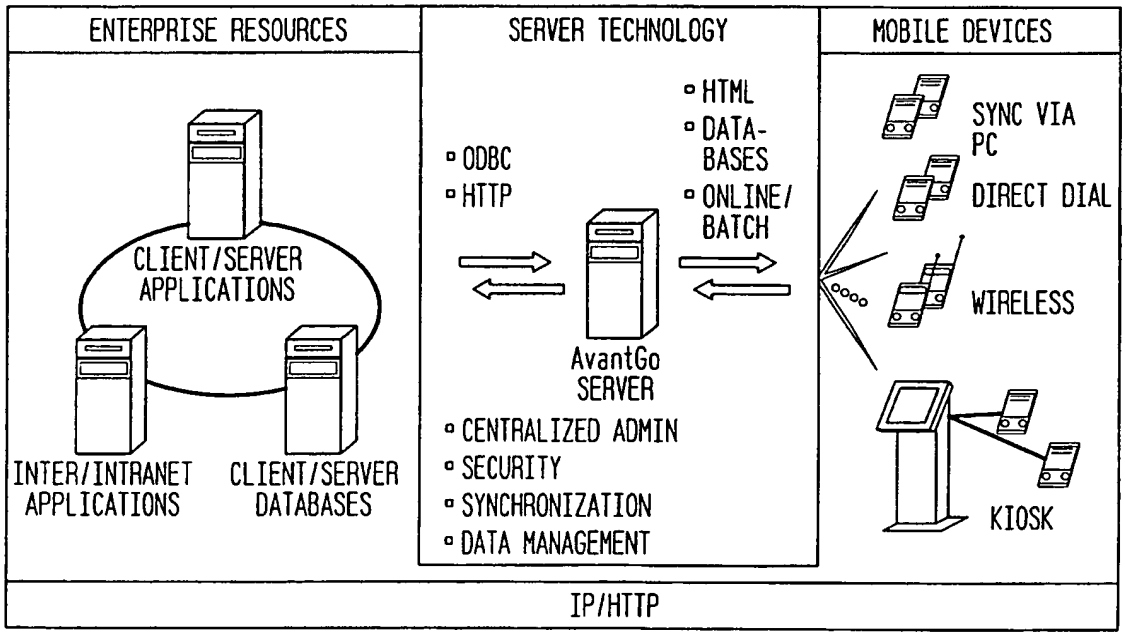

ENTERPRISE RESOURCES
SERVER TECHNOLOGY
MOBILE DEVICES
CLIENT/SERVER APPLICATIONS
INTER/INTRANET APPLICATIONS
CLIENT/SERVER DATABASES
ODBC
HTTP
HTML
DATA-BASES
ONLINE/ BATCH
AvantGo SERVER
CENTRALIZED ADMIN
SECURITY
SYNCHRONIZATION
DATA MANAGEMENT
SYNC VIA PC
DIRECT DIAL
WIRELESS
KIOSK
IP/HTTP

FIG. 31

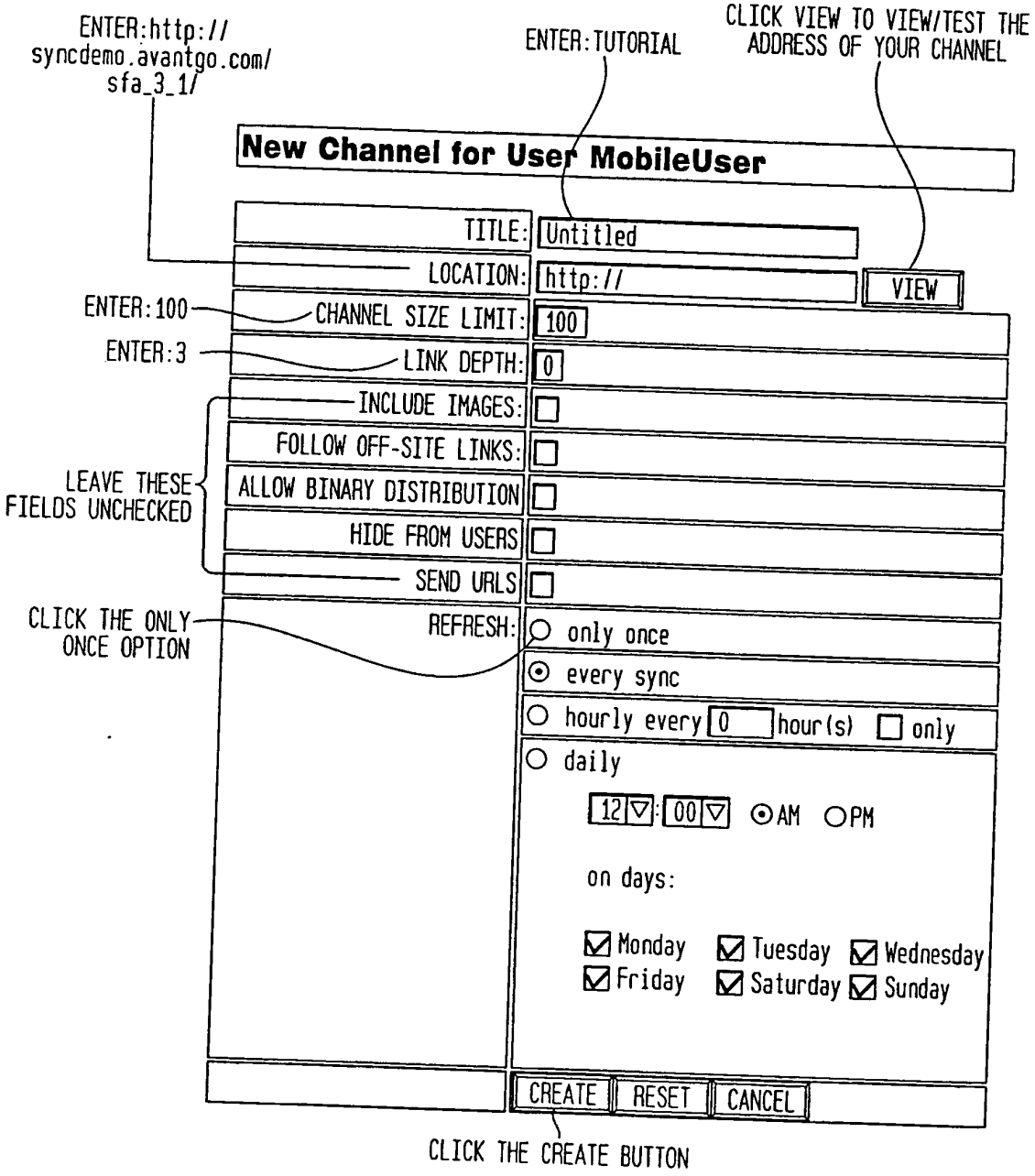

ENTER:http://
syncdemo.avantgo.com/
sfa_3_1/
ENTER:TUTORIAL
CLICK VIEW TO VIEW/TEST THE
ADDRESS OF YOUR CHANNEL
New Channel for User MobileUser
TITLE:
Untitled
LOCATION:
http://
VIEW
ENTER:100
CHANNEL SIZE LIMIT:
100
ENTER:3
LINK DEPTH:
0
INCLUDE IMAGES:
FOLLOW OFF-SITE LINKS:
LEAVE THESE
FIELDS UNCHECKED
ALLOW BINARY DISTRIBUTION
HIDE FROM USERS
SEND URLS
CLICK THE ONLY
ONCE OPTION
REFRESH:
only once
every sync
hourly every
0
hour(s)
only
daily
12
00
AM
PM
on days:
Monday
Tuesday
Wednesday
Friday
Saturday
Sunday
CREATE
RESET
CANCEL
CLICK THE CREATE BUTTON

FIG. 32

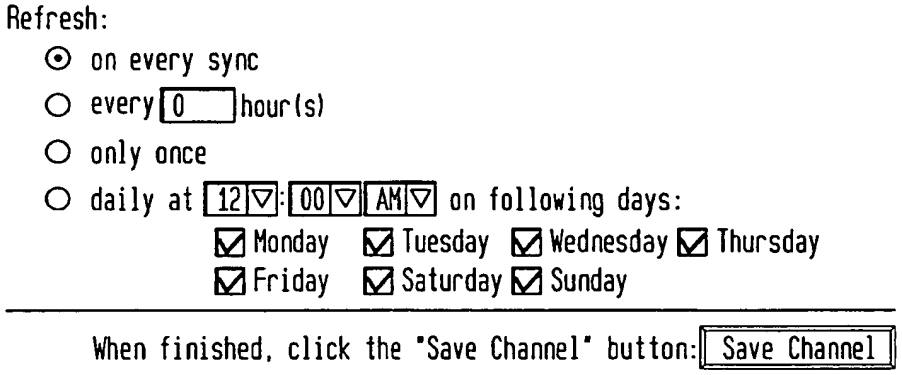

Refresh:
on every sync
every 0 hour(s)
only once
daily at 12 : 00 AM on following days:
Monday Tuesday Wednesday Thursday
Friday Saturday Sunday
When finished, click the "Save Channel" button:
Save Channel

FIG. 33

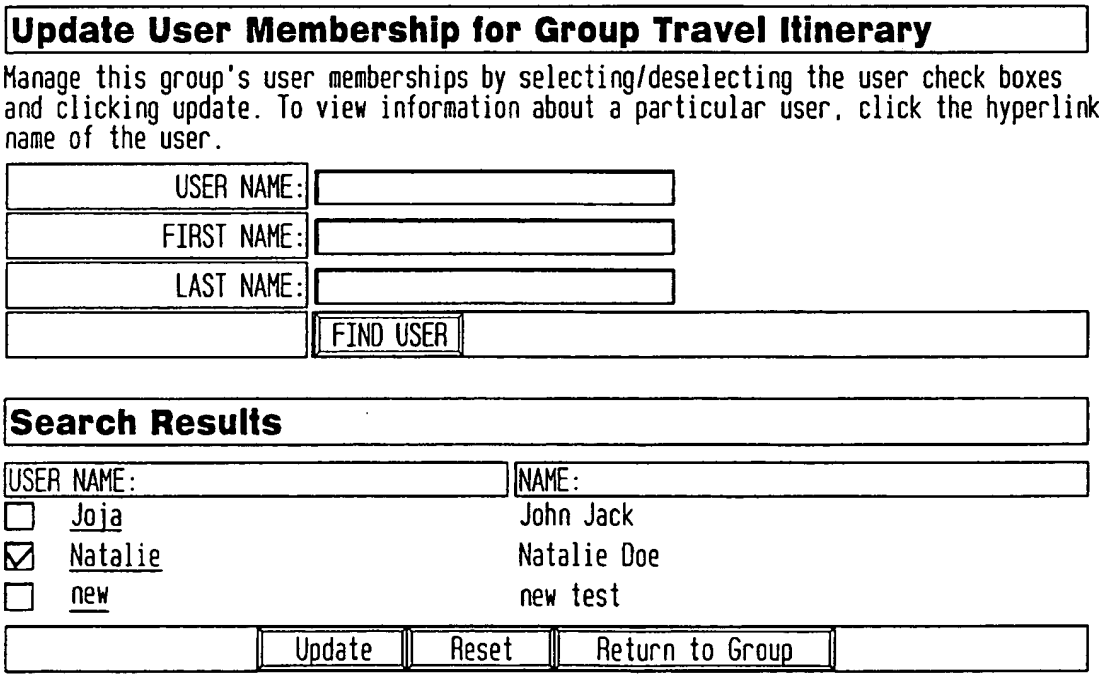

Update User Membership for Group Travel Itinerary
Manage this group's user memberships by selecting/deselecting the user check boxes and clicking update. To view information about a particular user, click the hyperlink name of the user.
USER NAME:
FIRST NAME:
LAST NAME:
FIND USER
Search Results
USER NAME:
NAME:
Joja
John Jack
Natalie
Natalie Doe
new
new test
Update
Reset
Return to Group

FIG. 34

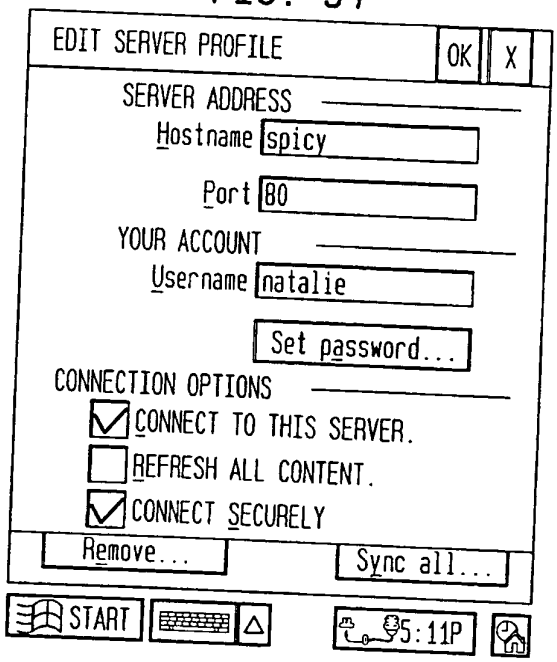

EDIT SERVER PROFILE
OK
X
SERVER ADDRESS
Hostname
spicy
Port
80
YOUR ACCOUNT
Username
natalie
Set password...
CONNECTION OPTIONS
CONNECT TO THIS SERVER.
REFRESH ALL CONTENT.
CONNECT SECURELY
Remove...
Sync all...
START
5:11P

FIG. 35

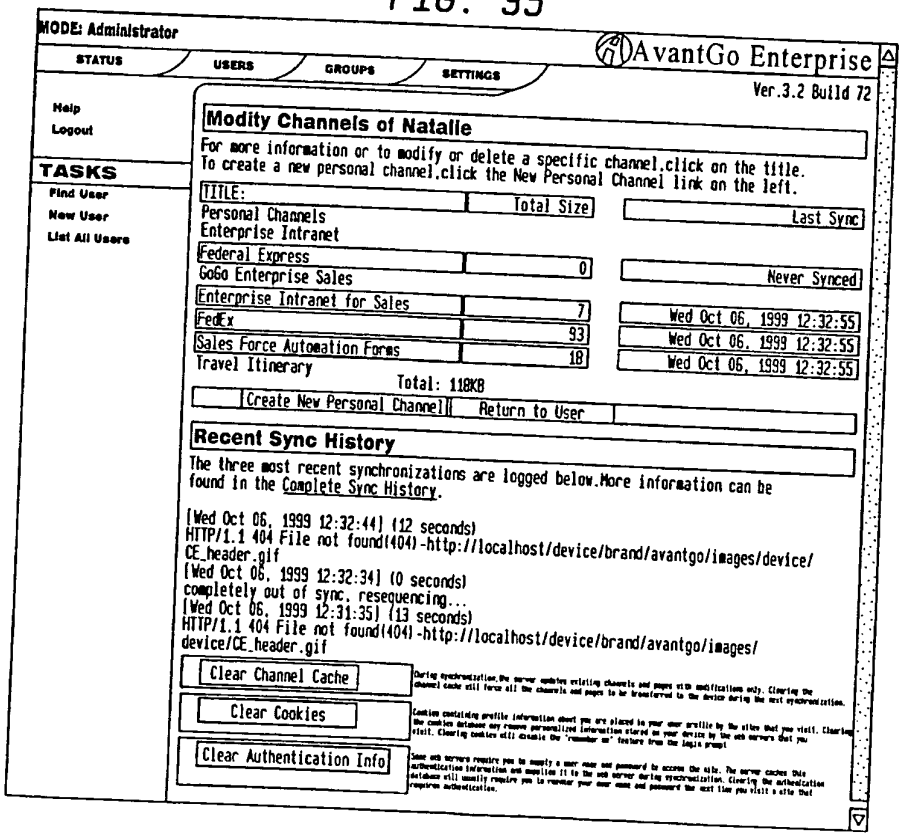

MODE: Administrator
AvantGo Enterprise
STATUS
USERS
GROUPS
SETTINGS
Ver.3.2 Build 72
Help
Logout
TASKS
Find User
New User
List All Users
Modify Channels of Natalie
For more information or to modify or delete a specific channel,click on the title.
To create a new personal channel,click the New Personal Channel link on the left.
TITLE:
Total Size
Last Sync
Personal Channels
Enterprise Intranet
Federal Express
0
Never Synced
GoGo Enterprise Sales
Enterprise Intranet for Sales
7
Wed Oct 06, 1999 12:32:55
FedEx
93
Wed Oct 06, 1999 12:32:55
Sales Force Automation Forms
18
Wed Oct 06, 1999 12:32:55
Travel Itinerary
Total: 118KB
Create New Personal Channel
Return to User
Recent Sync History
The three most recent synchronizations are logged below.More information can be found in the Complete Sync History.
[Wed Oct 06, 1999 12:32:44] (12 seconds)
HTTP/1.1 404 File not found(404)-http://localhost/device/brand/avantgo/images/device/CE_header.gif
[Wed Oct 06, 1999 12:32:34] (0 seconds)
completely out of sync, resequencing...
[Wed Oct 06, 1999 12:31:35] (13 seconds)
HTTP/1.1 404 File not found(404)-http://localhost/device/brand/avantgo/images/device/CE_header.gif
Clear Channel Cache
Clear Cookies
Clear Authentication Info

FIG. 36

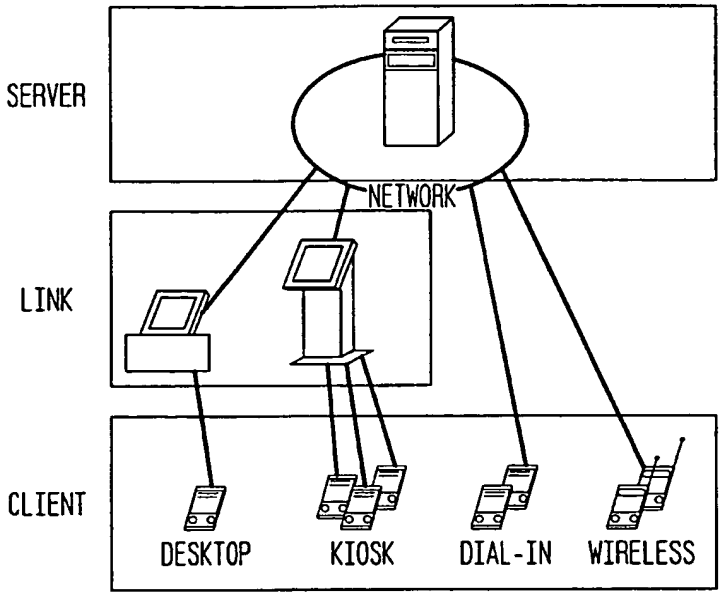

SERVER
NETWORK
LINK
CLIENT
DESKTOP
KIOSK
DIAL-IN
WIRELESS

FIG. 37

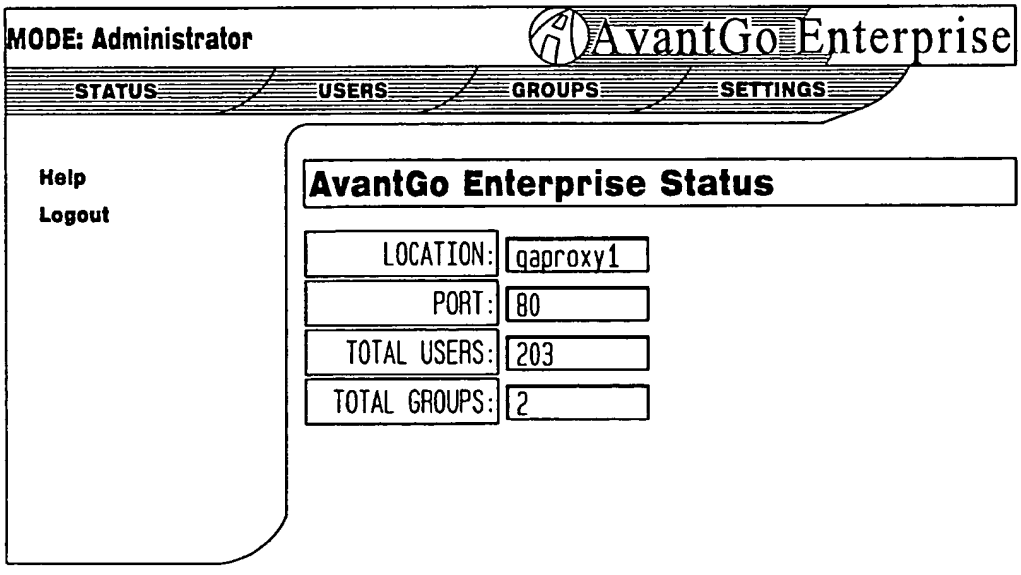

MODE: Administrator
AvantGo Enterprise
STATUS
USERS
GROUPS
SETTINGS
Help
Logout
AvantGo Enterprise Status
LOCATION: qaproxy1
PORT: 80
TOTAL USERS: 203
TOTAL GROUPS: 2

FIG. 38

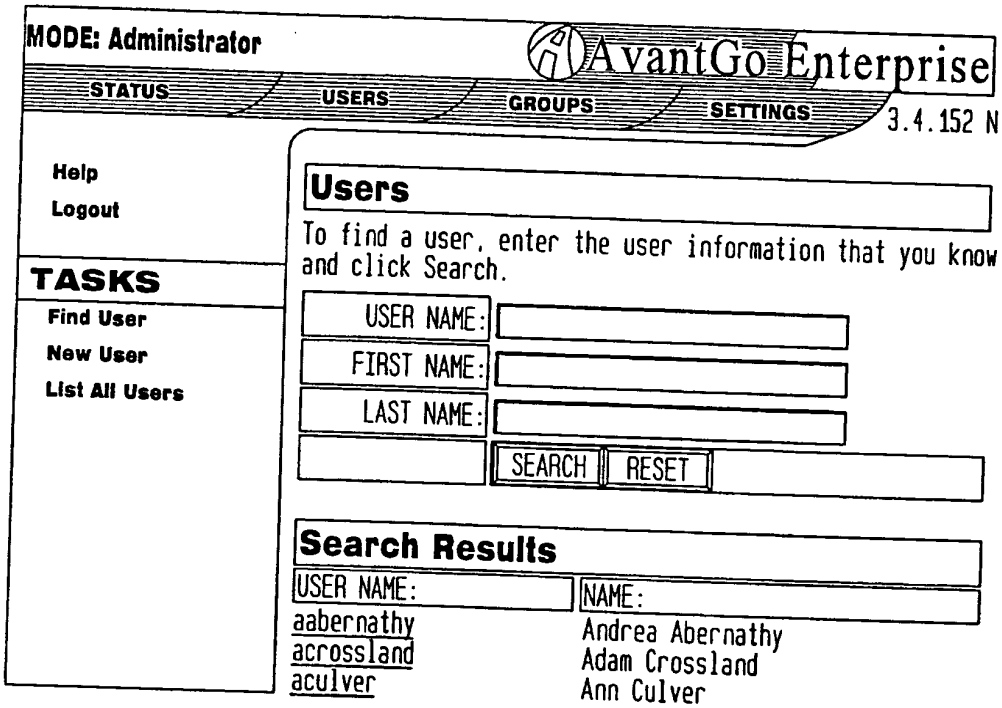

MODE: Administrator
AvantGo Enterprise
STATUS
USERS
GROUPS
SETTINGS
3.4.152 N
Help
Logout
TASKS
Find User
New User
List All Users
Users
To find a user, enter the user information that you know and click Search.
USER NAME:
FIRST NAME:
LAST NAME:
SEARCH
RESET
Search Results
USER NAME:
NAME:
aabernathy
Andrea Abernathy
acrossland
Adam Crossland
aculver
Ann Culver

FIG. 39

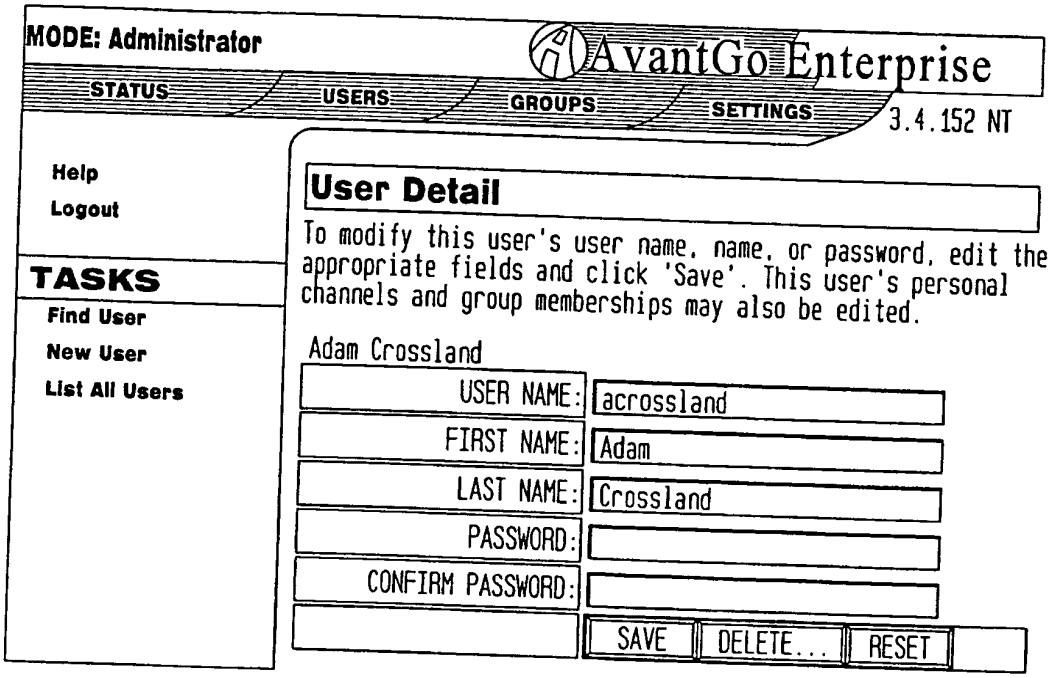

MODE: Administrator
AvantGo Enterprise
STATUS
USERS
GROUPS
SETTINGS
3.4.152 NT
Help
Logout
TASKS
Find User
New User
List All Users
User Detail
To modify this user's user name, name, or password, edit the appropriate fields and click 'Save'. This user's personal channels and group memberships may also be edited.
Adam Crossland
USER NAME:
acrossland
FIRST NAME:
Adam
LAST NAME:
Crossland
PASSWORD:
CONFIRM PASSWORD:
SAVE
DELETE...
RESET

FIG. 40

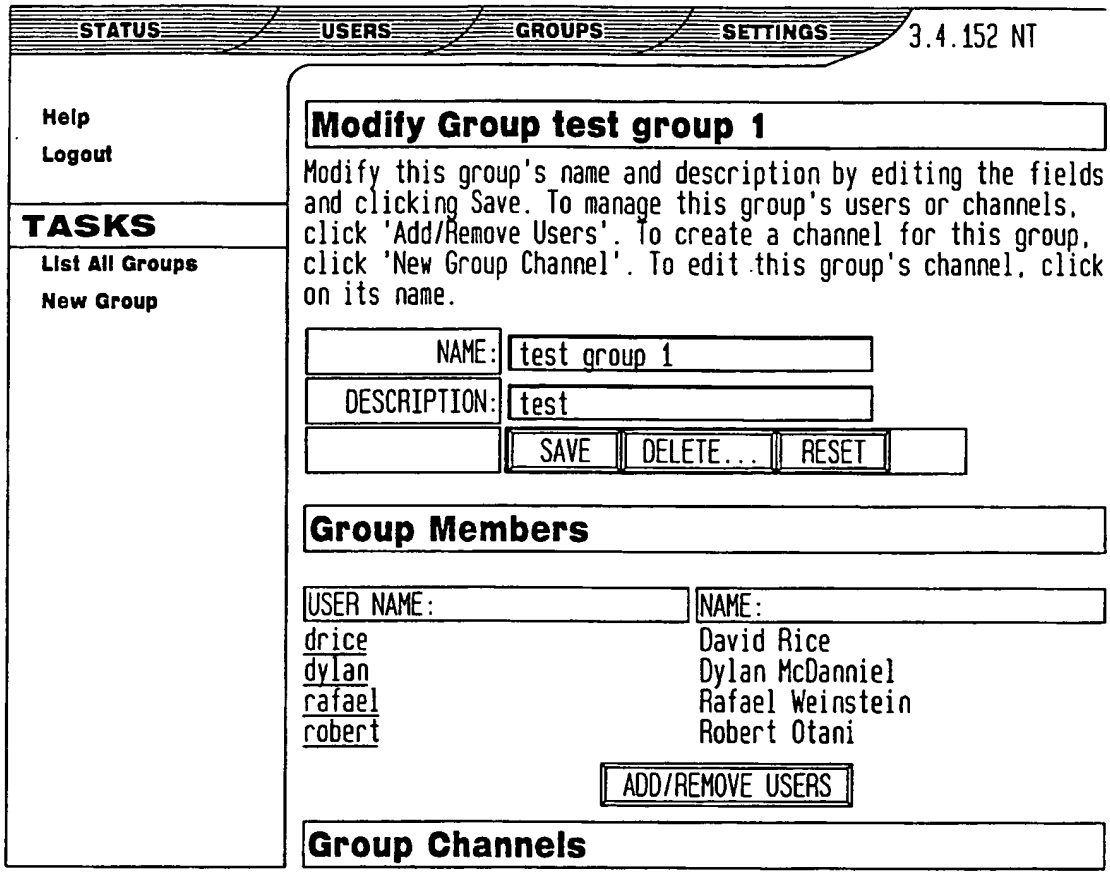

STATUS
USERS
GROUPS
SETTINGS
3.4.152 NT
Help
Logout
TASKS
List All Groups
New Group
Modify Group test group 1
Modify this group's name and description by editing the fields and clicking Save. To manage this group's users or channels, click 'Add/Remove Users'. To create a channel for this group, click 'New Group Channel'. To edit this group's channel, click on its name.
NAME: test group 1
DESCRIPTION: test
SAVE
DELETE...
RESET
Group Members
USER NAME:
NAME:
drice
David Rice
dylan
Dylan McDanniel
rafael
Rafael Weinstein
robert
Robert Otani
ADD/REMOVE USERS
Group Channels

FIG. 41

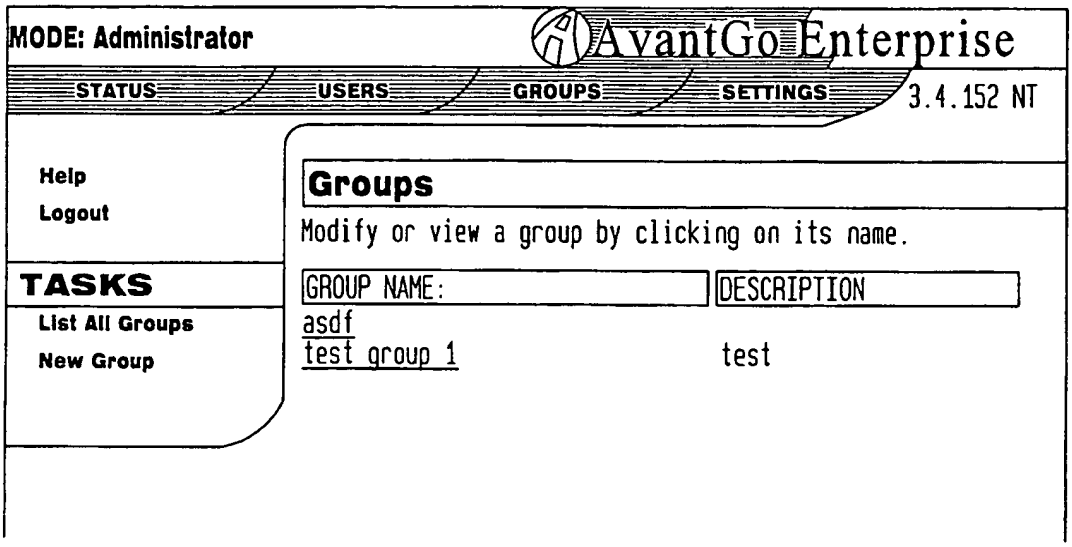

MODE: Administrator
AvantGo Enterprise
STATUS
USERS
GROUPS
SETTINGS
3.4.152 NT
Help
Logout
TASKS
List All Groups
New Group
Groups
Modify or view a group by clicking on its name.
GROUP NAME:
DESCRIPTION
asdf
test group 1
test

FIG. 42

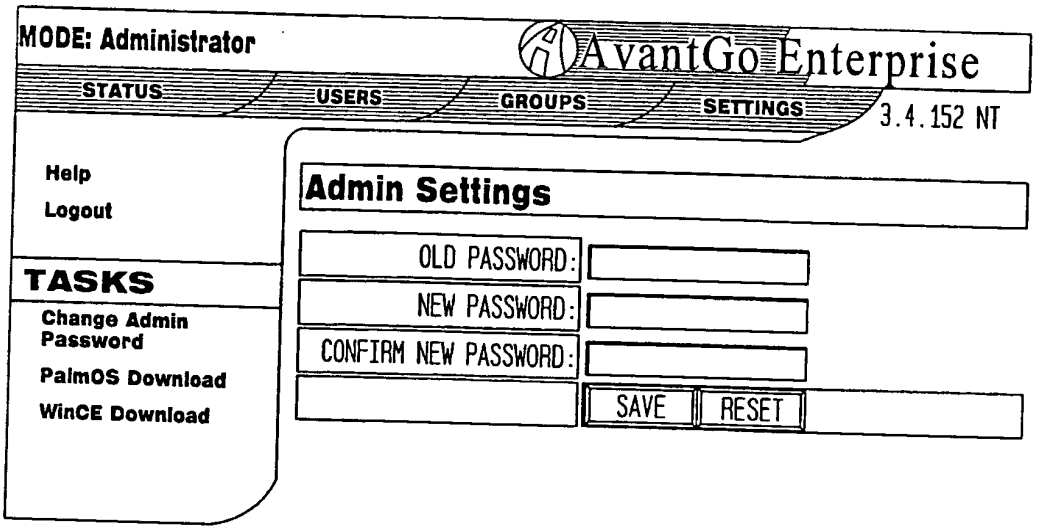

MODE: Administrator
AvantGo Enterprise
STATUS
USERS
GROUPS
SETTINGS
3.4.152 NT
Help
Logout
TASKS
Change Admin Password
PalmOS Download
WinCE Download
Admin Settings
OLD PASSWORD:
NEW PASSWORD:
CONFIRM NEW PASSWORD:
SAVE
RESET

FIG. 43

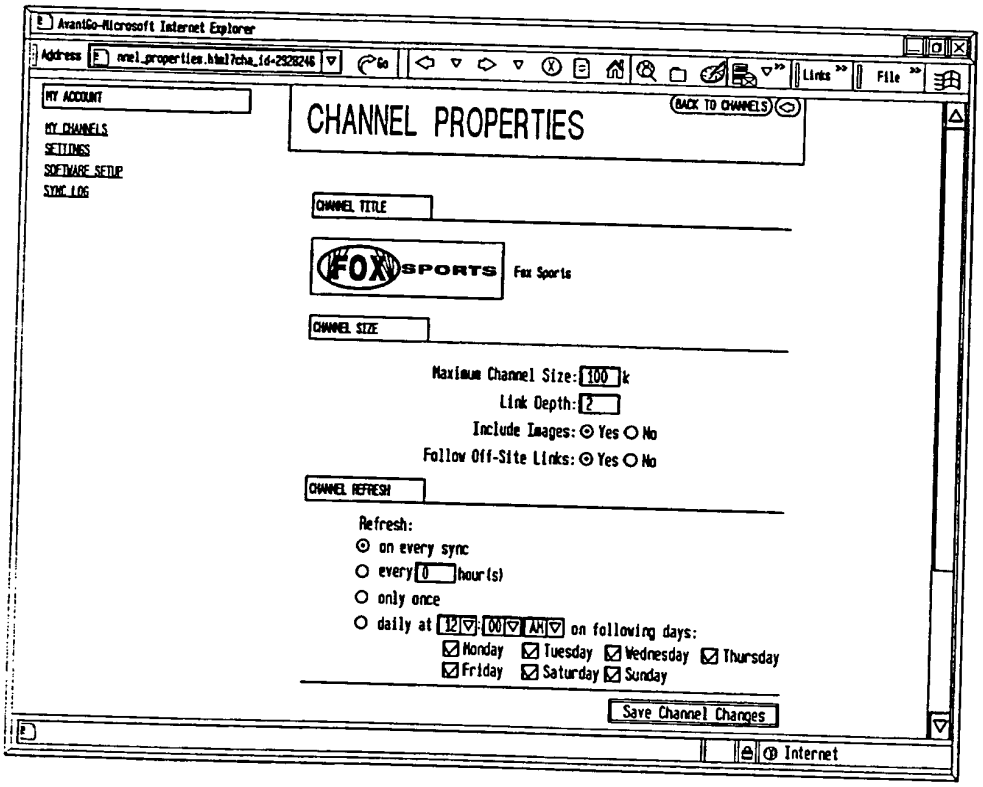

AvantGo-Microsoft Internet Explorer
MY ACCOUNT
MY CHANNELS
SETTINGS
SOFTWARE SETUP
SYNC LOG
CHANNEL PROPERTIES
BACK TO CHANNELS
CHANNEL TITLE
FOX SPORTS
Fox Sports
CHANNEL SIZE
Maximum Channel Size: 100 k
Link Depth: 2
Include Images: Yes No
Follow Off-Site Links: Yes No
CHANNEL REFRESH
Refresh:
on every sync
every 0 hour(s)
only once
daily at 12 00 AM on following days:
Monday Tuesday Wednesday Thursday
Friday Saturday Sunday
Save Channel Changes
Internet

FIG. 44

AvantGo TM

WELCOME, dkoehn68
Log out

MY ACCOUNT

MY CHANNELS
SETTINGS
SOFTWARE SETUP
SYNC LOG

AUTOCHANNEL™

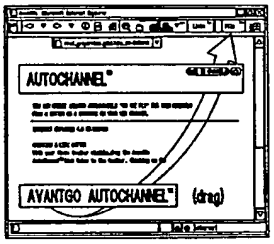
BACK TO CHANNELS

YOU CAN CREATE AVANTGO AUTOCHANNELS "ON THE FLY" FOR YOUR HANDHELD FROM A BUTTON OR A BOOKMARK ON YOUR WEB BROWSER.

INTERNET EXPLORER 4.0 OR HIGHER

CREATING A LINK BUTTON
With your links toolbar visible, drag the AvantGo AutoChannel™ link below to the toolbar. Clicking on the button while browsing a website will let you add that website to your handheld device.

Drag this link: AvantGo AutoChannel



CREATING A BOOKMARK
Right-click on AvantGo AutoChannel™ link below and select "Add to Favorites...". Choosing this in your favorites will let you add the website you are browsing as an AvantGo channel on your handheld.

Right-click this link: AvantGo AutoChannel

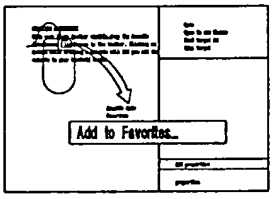


FIG. 45

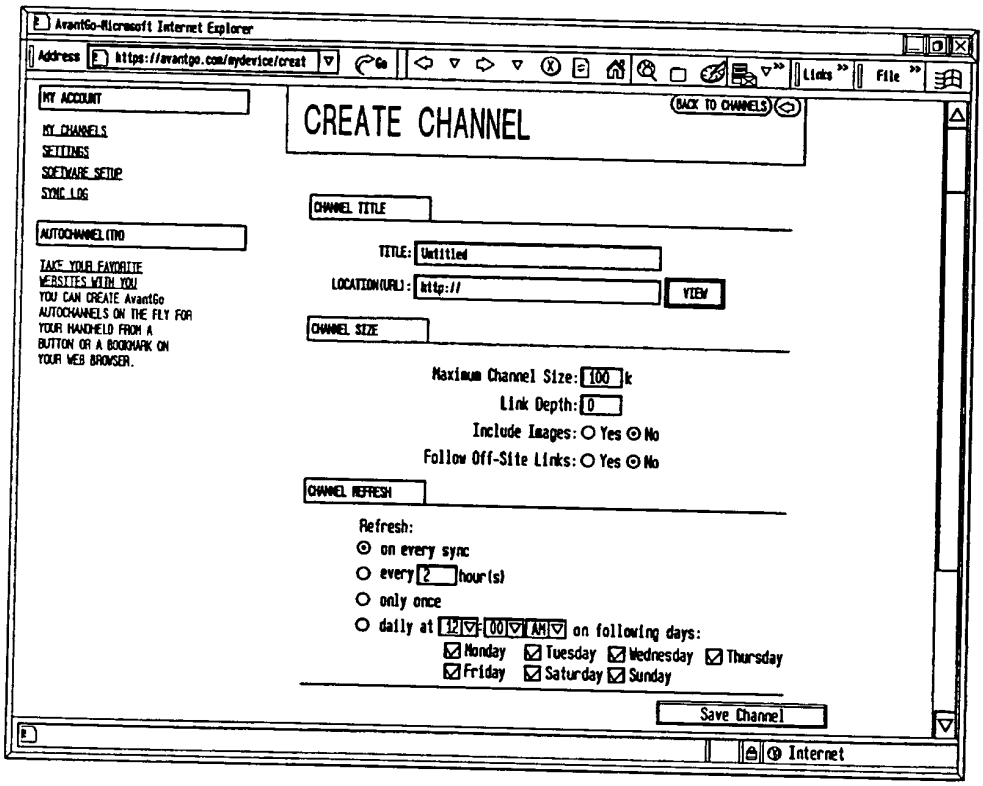

AvantGo-Microsoft Internet Explorer
Address https://avantgo.com/mydevice/creat
Go
Links
File
MY ACCOUNT
MY CHANNELS
SETTINGS
SOFTWARE SETUP
SYNC LOG
AUTOCHANNEL (TM)
TAKE YOUR FAVORITE WEBSITES WITH YOU! YOU CAN CREATE AvantGo AUTOCHANNELS ON THE FLY FOR YOUR HANDHELD FROM A BUTTON OR A BOOKMARK ON YOUR WEB BROWSER.
CREATE CHANNEL
BACK TO CHANNELS
CHANNEL TITLE
TITLE: Untitled
LOCATION(URL): http://
VIEW
CHANNEL SIZE
Maximum Channel Size: 100 k
Link Depth: 0
Include Images: Yes No
Follow Off-Site Links: Yes No
CHANNEL REFRESH
Refresh:
on every sync
every 2 hour(s)
only once
daily at 12 : 00 AM on following days:
Monday Tuesday Wednesday Thursday
Friday Saturday Sunday
Save Channel
Internet

FIG. 46

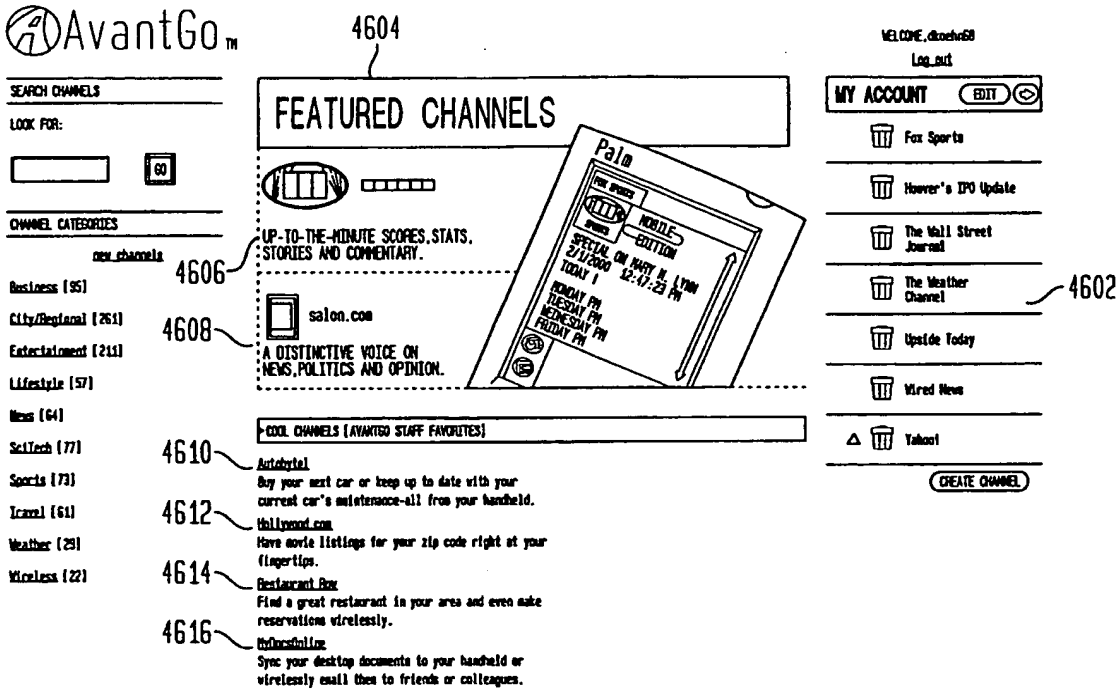

AvantGo™
4604
FEATURED CHANNELS
SEARCH CHANNELS
LOOK FOR:
GO
CHANNEL CATEGORIES
new channels
Business [95]
City/Regional [261]
Entertainment [215]
Lifestyle [57]
News [64]
SciTech [77]
Sports [73]
Travel [61]
Weather [29]
Wireless [22]
4606
UP-TO-THE-MINUTE SCORES, STATS, STORIES AND COMMENTARY.
4608
salon.com
A DISTINCTIVE VOICE ON NEWS, POLITICS AND OPINION.
COOL CHANNELS [AVANTGO STAFF FAVORITES]
4610
Autobytel
Buy your next car or keep up to date with your current car's maintenance-all from your handheld.
4612
Hollywood.com
Have movie listings for your zip code right at your fingertips.
4614
Restaurant Row
Find a great restaurant in your area and even make reservations wirelessly.
4616
MyDocsOnline
Sync your desktop documents to your handheld or wirelessly email them to friends or colleagues.
WELCOME, dkoehn68
Log out
MY ACCOUNT
EDIT
Fox Sports
Hoover's IPO Update
The Wall Street Journal
The Weather Channel
4602
Upside Today
Wired News
Yahoo!
CREATE CHANNEL

FIG. 47

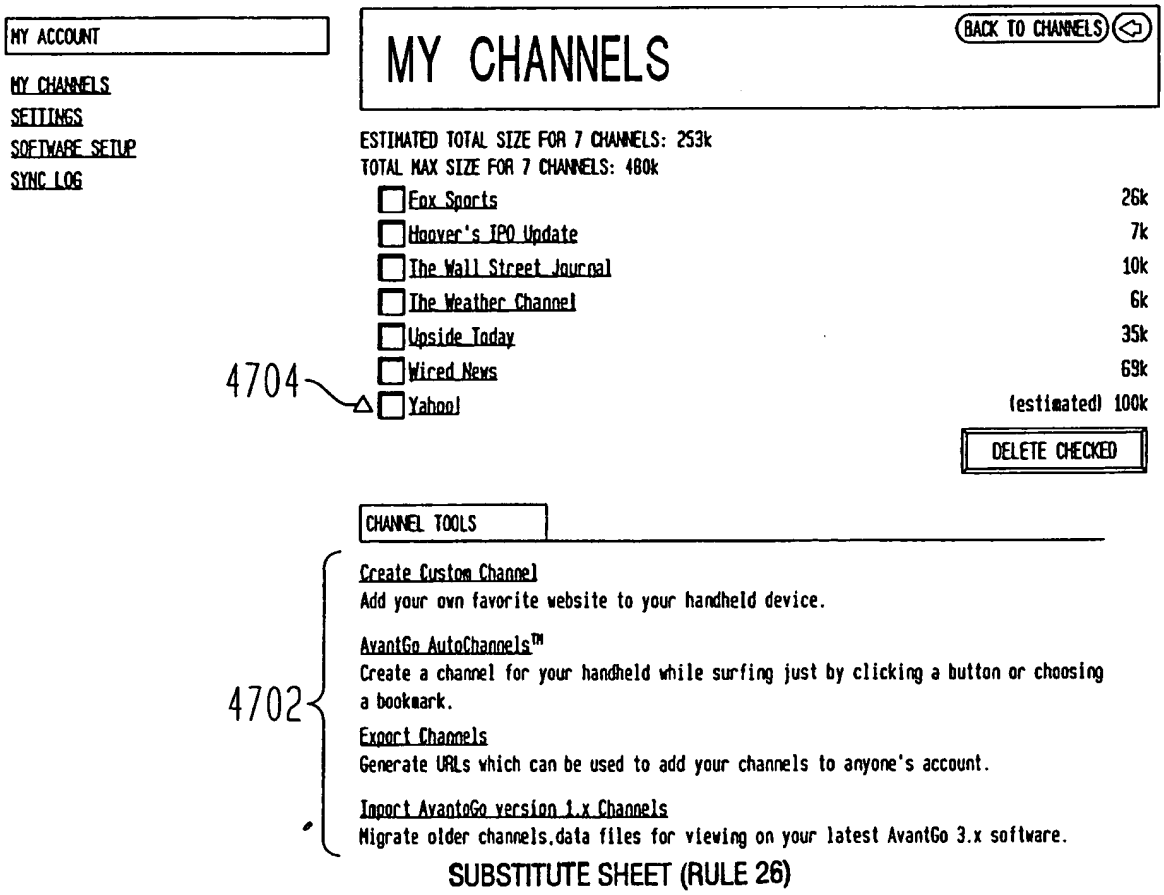

MY ACCOUNT
MY CHANNELS
SETTINGS
SOFTWARE SETUP
SYNC LOG
MY CHANNELS
BACK TO CHANNELS
ESTIMATED TOTAL SIZE FOR 7 CHANNELS: 253k
TOTAL MAX SIZE FOR 7 CHANNELS: 480k
Fox Sports 26k
Hoover's IPO Update 7k
The Wall Street Journal 10k
The Weather Channel 6k
Upside Today 35k
Wired News 69k
4704
Yahoo! (estimated) 100k
DELETE CHECKED
CHANNEL TOOLS
4702
Create Custom Channel
Add your own favorite website to your handheld device.
AvantGo AutoChannels™
Create a channel for your handheld while surfing just by clicking a button or choosing a bookmark.
Export Channels
Generate URLs which can be used to add your channels to anyone's account.
Import AvantGo version 1.x Channels
Migrate older channels.data files for viewing on your latest AvantGo 3.x software.

FIG. 48

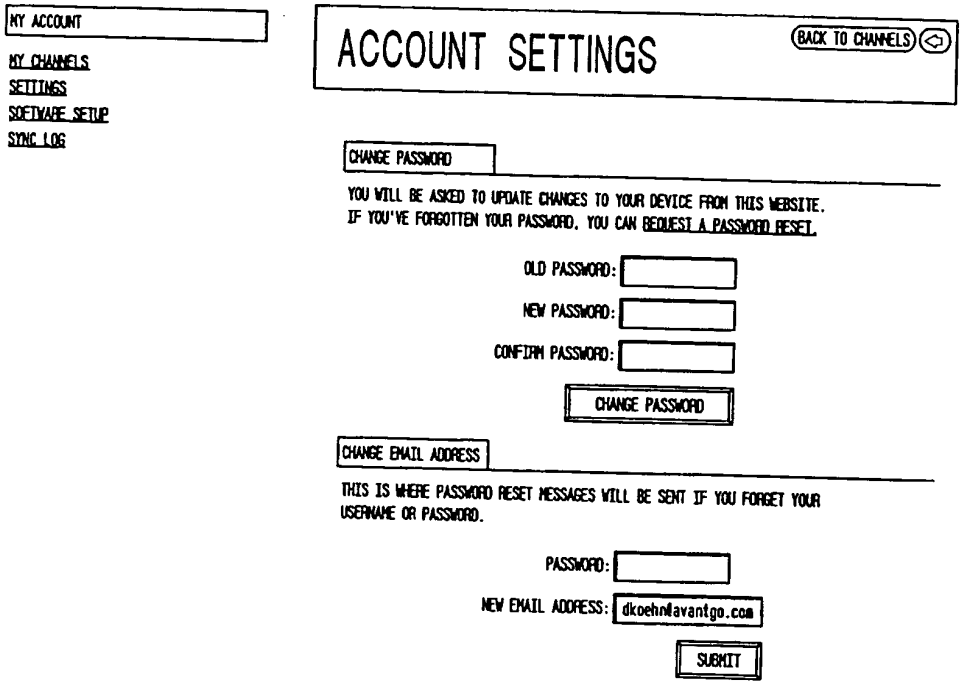

MY ACCOUNT
MY CHANNELS
SETTINGS
SOFTWARE SETUP
SYNC LOG
ACCOUNT SETTINGS
BACK TO CHANNELS
CHANGE PASSWORD
YOU WILL BE ASKED TO UPDATE CHANGES TO YOUR DEVICE FROM THIS WEBSITE.
IF YOU'VE FORGOTTEN YOUR PASSWORD, YOU CAN REQUEST A PASSWORD RESET.
OLD PASSWORD:
NEW PASSWORD:
CONFIRM PASSWORD:
CHANGE PASSWORD
CHANGE EMAIL ADDRESS
THIS IS WHERE PASSWORD RESET MESSAGES WILL BE SENT IF YOU FORGET YOUR USERNAME OR PASSWORD.
PASSWORD:
NEW EMAIL ADDRESS:
dkoehn@avantgo.com
SUBMIT

FIG. 49

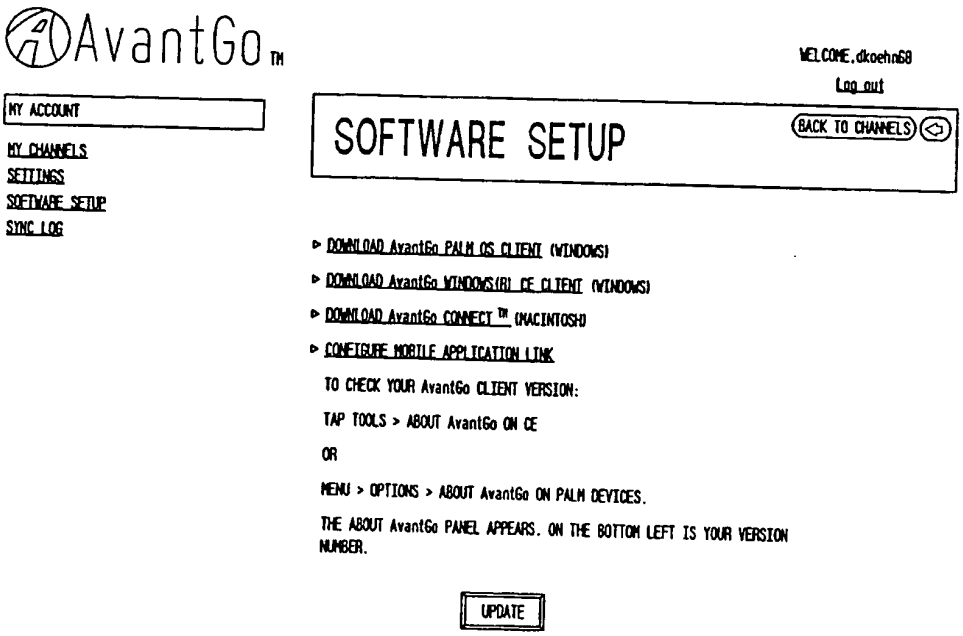

AvantGo TM
WELCOME, dkoehn68
Log out
MY ACCOUNT
MY CHANNELS
SETTINGS
SOFTWARE SETUP
SYNC LOG
SOFTWARE SETUP
BACK TO CHANNELS
DOWNLOAD AvantGo PALM OS CLIENT (WINDOWS)
DOWNLOAD AvantGo WINDOWS(R) CE CLIENT (WINDOWS)
DOWNLOAD AvantGo CONNECT TM (MACINTOSH)
CONFIGURE MOBILE APPLICATION LINK
TO CHECK YOUR AvantGo CLIENT VERSION:
TAP TOOLS > ABOUT AvantGo ON CE
OR
MENU > OPTIONS > ABOUT AvantGo ON PALM DEVICES.
THE ABOUT AvantGo PANEL APPEARS. ON THE BOTTOM LEFT IS YOUR VERSION NUMBER.
UPDATE

FIG. 50

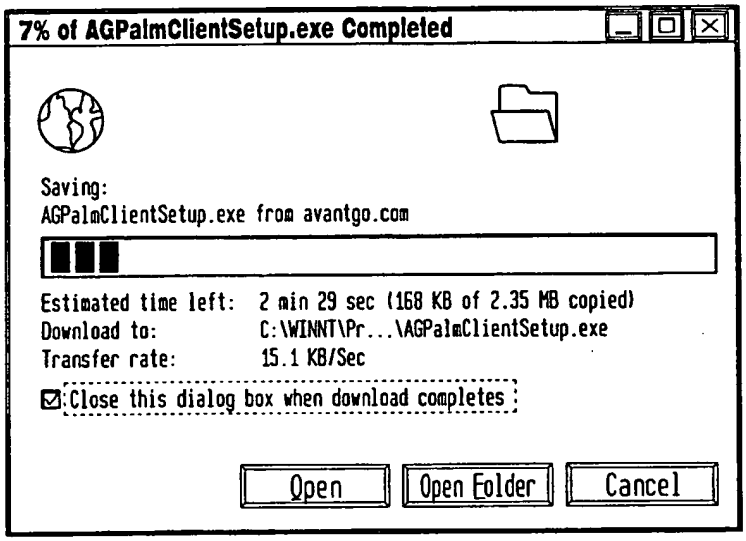

7% of AGPalmClientSetup.exe Completed
Saving:
AGPalmClientSetup.exe from avantgo.com
Estimated time left: 2 min 29 sec (168 KB of 2.35 MB copied)
Download to: C:\WINNT\Pr...\AGPalmClientSetup.exe
Transfer rate: 15.1 KB/Sec
Close this dialog box when download completes
Open
Open Folder
Cancel

FIG. 51

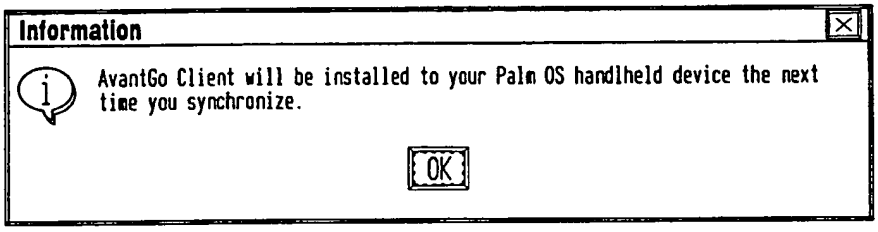

Information
AvantGo Client will be installed to your Palm OS handheld device the next time you synchronize.
OK

FIG. 52

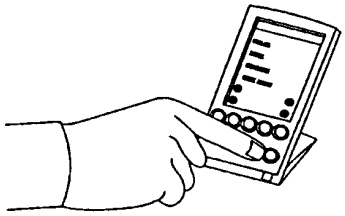
AvantGo™

SETUP CHECKLIST
✓DOWNLOAD
▸INSTALL
SYNCHRONIZE
CREATE ACCOUNT
CONFIGURE
SYNCHRONIZE

SYNCHRONIZE

Place your device in its cradle and press the HotSync button.

When you synchronize, AvantGo 3.x will be installed on your device. After synchronization is complete, click the "Next" button at the bottom of the page.



NEXT >>

FIG. 53

AvantGo™

SETUP CHECKLIST
✓DOWNLOAD
✓INSTALL
✓SYNCHRONIZE
▸CREATE ACCOUNT
CONFIGURE
SYNCHRONIZE

USE EXISTING ACCOUNT?

Select below to use this account or to create a new one

Username: dkoehn68
Email: dkoehn@avantgo.com

◉ Use this account
○ Create a new account

NEXT >>

FIG. 54

AvantGo™

SETUP CHECKLIST
✓DOWNLOAD
✓INSTALL
✓SYNCHRONIZE
✓CREATE ACCOUNT
▸CONFIGURE
SYNCHRONIZE

CONFIGURE MOBILE LINK

Click the link below to configure Mobile Link™

By clicking below, you will ensure your AvantGo 3.x software is communicating properly with AvantGo's servers.

CLICK HERE TO CONFIGURE 5402

NEXT >>

FIG. 55

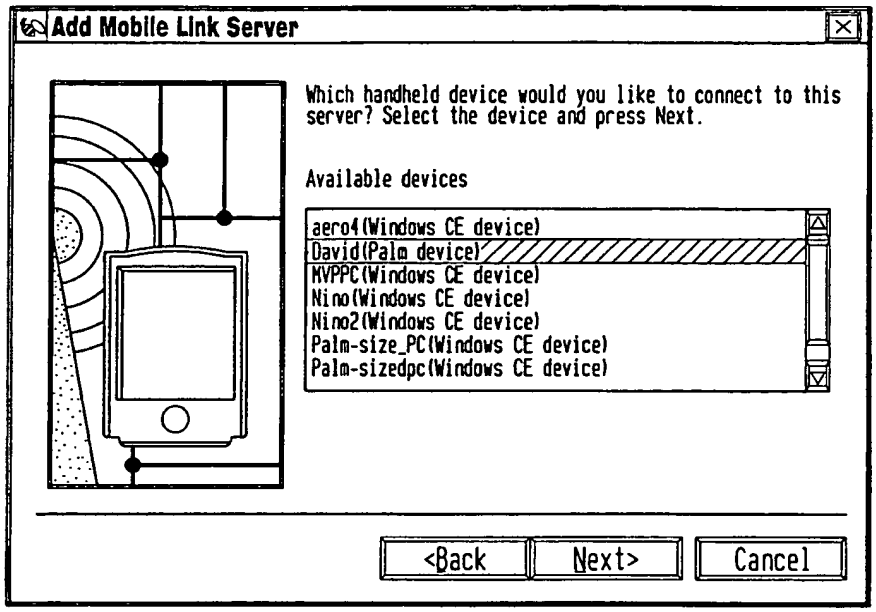

Add Mobile Link Server
Which handheld device would you like to connect to this server? Select the device and press Next.
Available devices
aero4(Windows CE device)
David(Palm device)
MVPPC(Windows CE device)
Nino(Windows CE device)
Nino2(Windows CE device)
Palm-size_PC(Windows CE device)
Palm-sizedpc(Windows CE device)
<Back
Next>
Cancel

FIG. 56

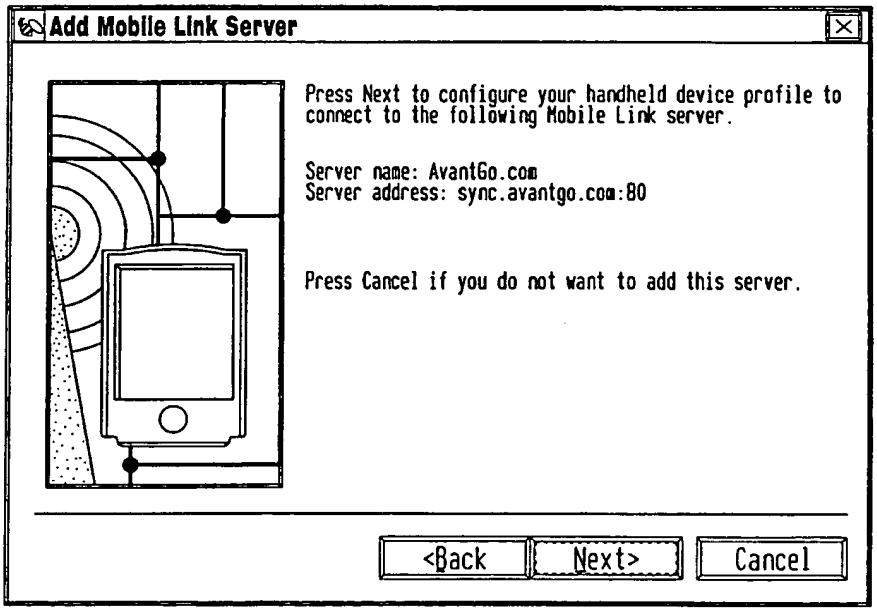

Add Mobile Link Server
Press Next to configure your handheld device profile to connect to the following Mobile Link server.
Server name: AvantGo.com
Server address: sync.avantgo.com:80
Press Cancel if you do not want to add this server.
<Back
Next>
Cancel

FIG. 57

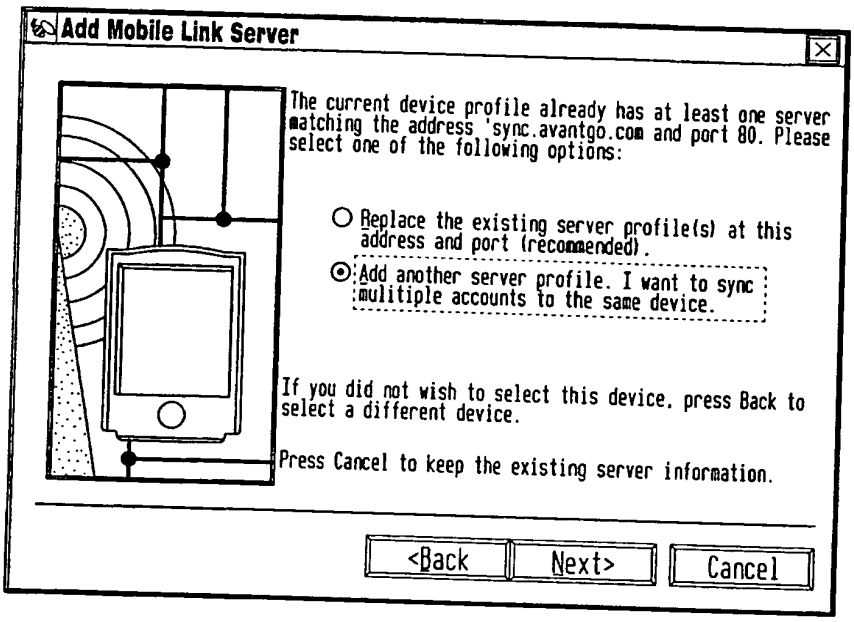

Add Mobile Link Server
The current device profile already has at least one server matching the address 'sync.avantgo.com and port 80. Please select one of the following options:
Replace the existing server profile(s) at this address and port (recommended).
Add another server profile. I want to sync mulitiple accounts to the same device.
If you did not wish to select this device, press Back to select a different device.
Press Cancel to keep the existing server information.
<Back
Next>
Cancel

FIG. 58

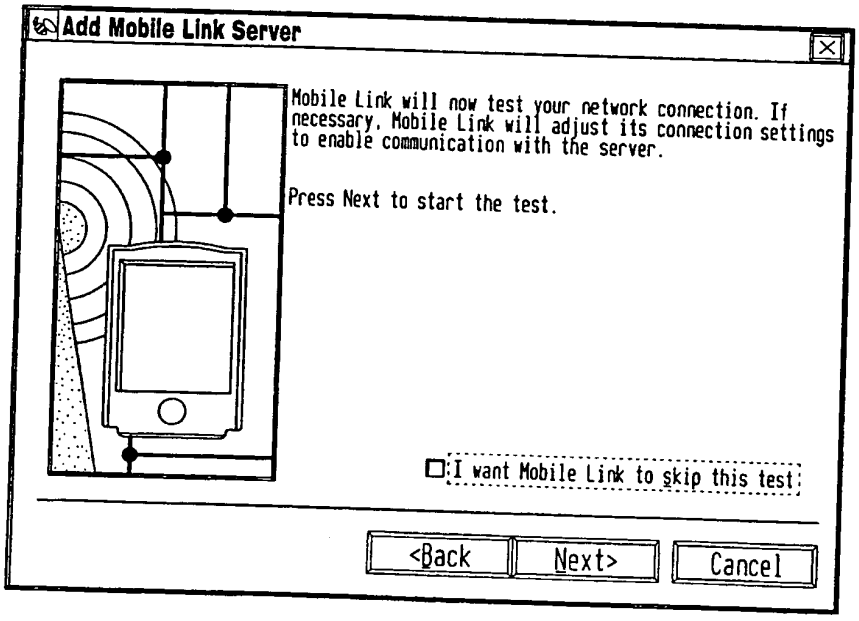

Add Mobile Link Server
Mobile Link will now test your network connection. If necessary, Mobile Link will adjust its connection settings to enable communication with the server.
Press Next to start the test.
I want Mobile Link to skip this test
<Back
Next>
Cancel

FIG. 59

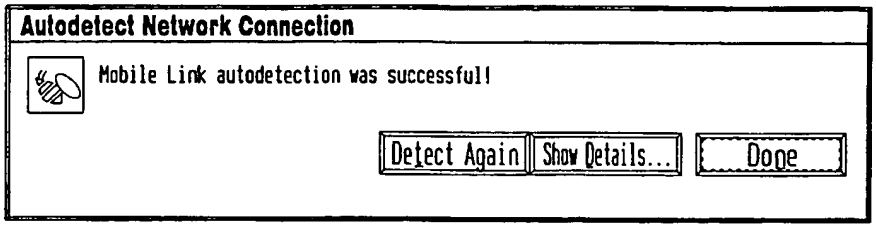

Autodetect Network Connection
Mobile Link autodetection was successful!
Detect Again
Show Details...
Done

FIG. 60

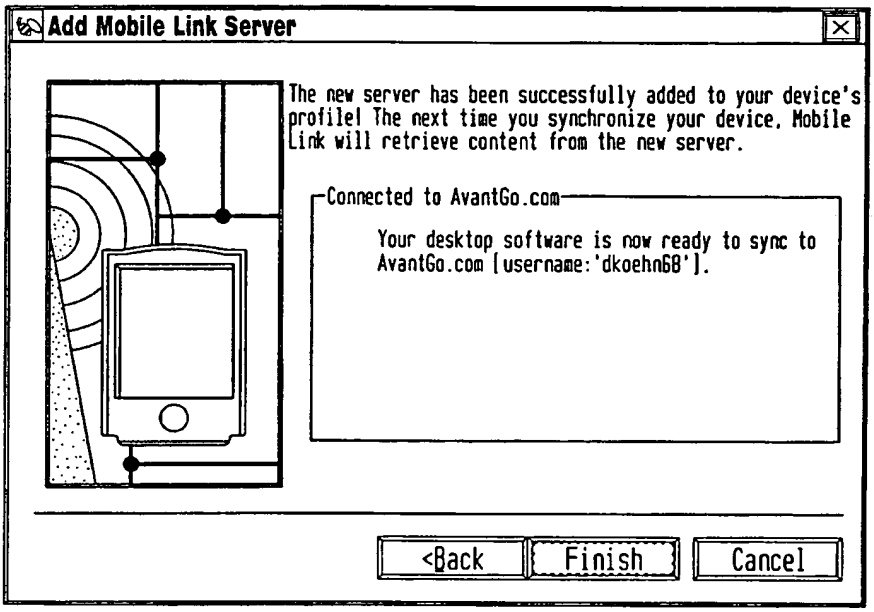

Add Mobile Link Server
The new server has been successfully added to your device's profile! The next time you synchronize your device, Mobile Link will retrieve content from the new server.
Connected to AvantGo.com
Your desktop software is now ready to sync to AvantGo.com [username:'dkoehn68'].
<Back
Finish
Cancel

FIG. 61

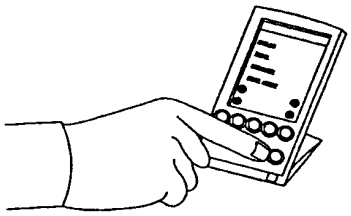
AvantGo™

SETUP CHECKLIST
✓DOWNLOAD
✓INSTALL
✓SYNCHRONIZE
✓CREATE ACCOUNT
✓CONFIGURE
•SYNCHRONIZE

SYNCHRONIZE

Place your device in its cradle and press the HotSync button.

When you synchronize, your AvantGo 3.x software will connect to the Avantgo servers for the first time. After synchronization is complete, click the "Next" button at the bottom of the page.



NEXT >>

FIG. 62

AvantGo™

SETUP CHECKLIST
✓DOWNLOAD
✓INSTALL
✓SYNCHRONIZE
✓CREATE ACCOUNT
✓CONFIGURE
✓SYNCHRONIZE

CONGRATULATIONS!

You are now an AvantGo user

Pick up your device, click on the "applications" button and select AvantGo to begin navigating channels. Use the links below to begin navigating AvantGo.

CHANNELS
Choose from 1000s of exciting channels from leading providers

ACCOUNT
View and edit your personalized account settings

FINISH >>

FIG. 63A

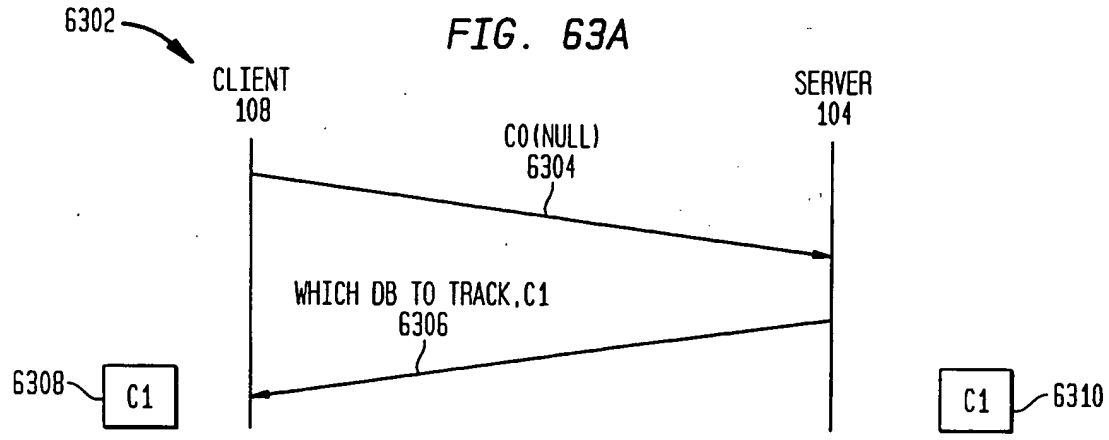
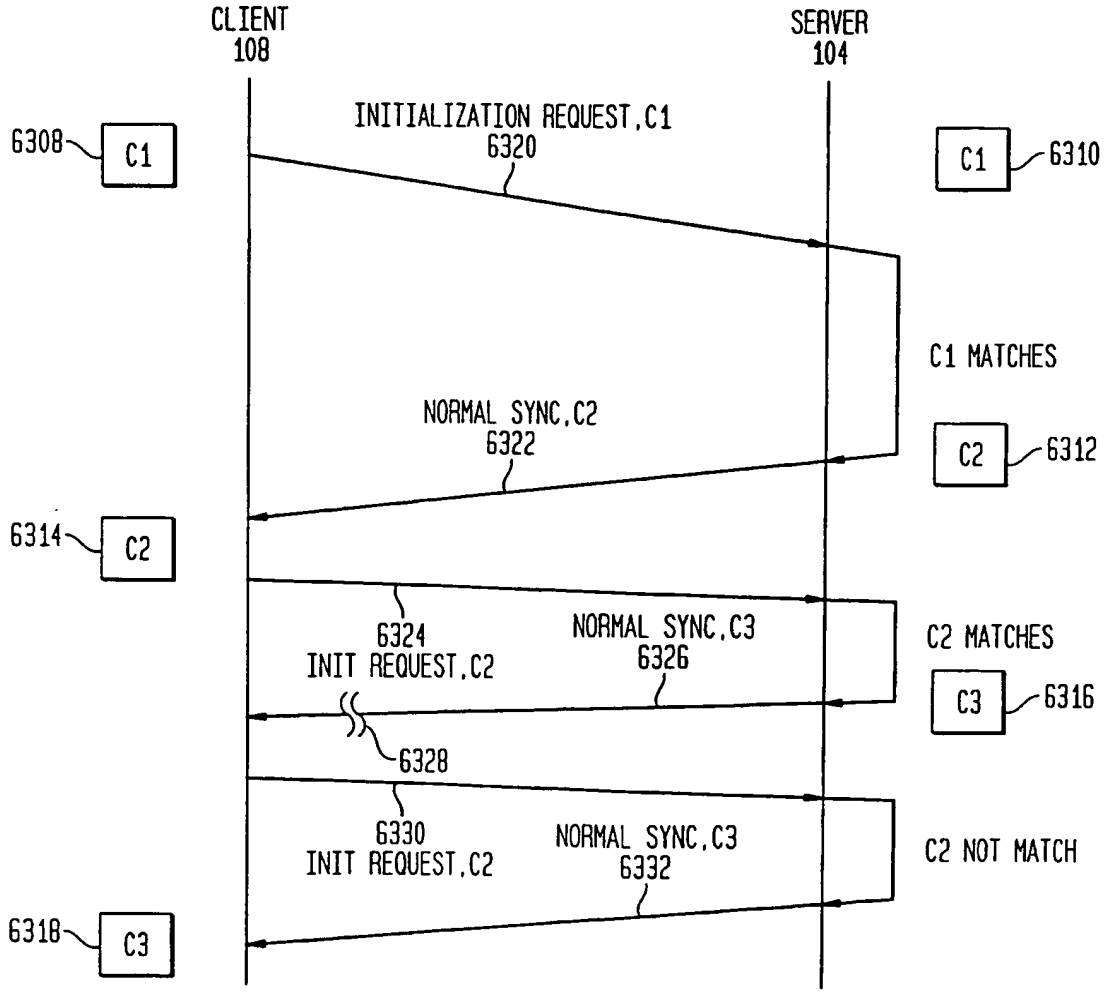


FIG. 63B



THIS PAGE BLANK (USPTO)